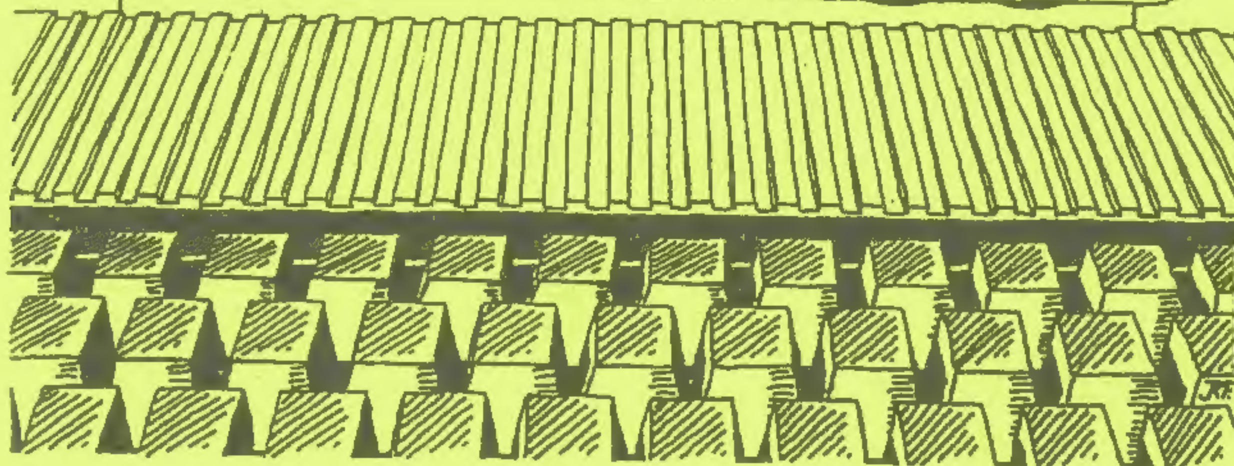
Atom Nieuws

Jaargang
nummer





Bestuur:

Voorzitter:	Secretaris:	Penningmeester:
N.Stad	J.Hartog	T.Rutten
Plataanweg 47	Keyenbergseweg 60	Berkenlaan 24
1544 PB Zaandijk	6871 WK Renkum	3737 RN Groenekan
Tel.075-280808	Tel.08373-13757	Tel.03461-3495

Clubwinkel:	Hardware cie.:	Redactie Atom Nieuws:
P.Grevelt	Y.Tuk	H. de Ruiter
Emmastraat 22	Postbus 257	Polarisstraat 25
1782 PD Den Helder	2980 AG Ridderkerk	8303 AC Emmeloord
Tel.02230-23453		

Contributie 1987: fl. 60.00
Leden buitenland: fl. 75.00

Rekeningnummers Atom Computer Club:
Giro 5244293

Redactie Atom Nieuws:
Joop Ballijns
Harry de Ruiter

Ledenadministratie:	Datasheets:
S.van Leeuwen	G.Akkerman
Kompasstraat 32	Wikke 1
1973 PX IJmuiden	1273 BR Huizen
Tel.02550-22435	Tel.02152-60294

Uiterste datum inlevering copy:
nr. 87-4 06-07-1987

De clubwinkel:

80-koloms videokaart. incl.alle onderdelen behalve de RAM-IC's. bouwpakket	fl.130.00
80-koloms videokaart,gebouwd en getest,excl.RAM-IC's	fl.180.00
Geheugenkaart: 16 kByte,excl.onderdelen	fl. 35.00
Schakelkaart: meerdere EPROM's op Axxx	fl. 47.50
Minischakelkaart	fl. 16.00
Z-80 kaart:CP/M,excl. onderdelen	fl. 85.00
Herdruk ACORN NIEUWS 1982; 97 pag.wetenswaardigheden	fl. 6.00
ATOM NIEUWS jaargang 1983, ruim 450 pag.	fl. 30.00
ATOM NIEUWS jaargang 1984. ruim 650 pag.	fl. 35.00
ATOM NIEUWS jaargang 1985, ruim 650 pag.	fl. 35.00
ATOM NIEUWS Jaargang 1986, ruim 500 pag.	fl. 35.00
ATOM-WARE 1: Basic interpreter van de ATOM, 98 pag.	fl. 6.00
ATOM-WARE 2: ATOM Disc Operating System,68 pag.	fl. 5.00
ATOM-WARE 3: ATOM Monitor Operating System,80 pag.	fl. 5.00

Levering;
Bij uw reg.penningmeester, eventueel rechtstreeks bij de fed. penningmeester.Bij rechtstreekse bestelling dient u het bedrag van het gewenste artikel te storten op de giro van de federatie onder vermelding van de naam van het artikel en uw lidnummer.Uw betaling dient vermeerderd te zijn met fl. 4.00 voor portokosten.

ATOM NIEUWS is een uigave van de federatie ATOM computerclubs Ned/Belgie en verschijnt 6 - 8 keer per jaar.
De redactie gaat er vanuit dat de ingezonden copy gemaakt is door de inzender tenzij in de publicatie uitdrukkelijk anders is vermeld.De aansprakelijkheid echter betreffend de auteursrechten ligt zonder enig voorbehoud volledig bij de inzender.

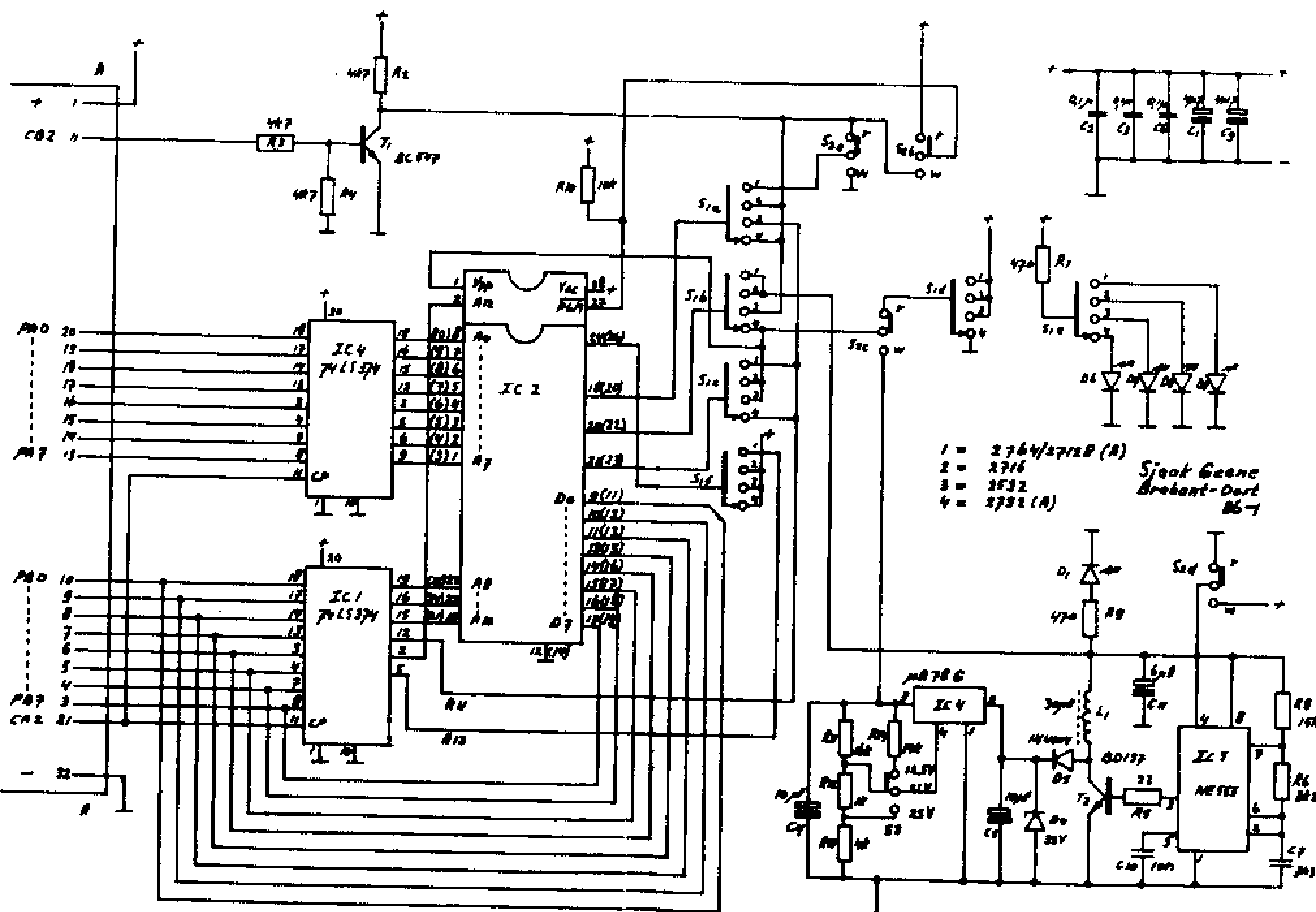
	PROM87	002900	00AFAF	001091	013
	SPRITES	002900	00AFAF	000515	03D
	ZEESLAG	002900	00AFAF	003800	005
Regioschijf 6-3	POKE	002900	00AFAF	000261	002
	MDCR4	002900	00C2B2	001DD3	0AA
	MDCR3	002900	00C2B2	0011FB	098
	MDCR2	002900	00C2B2	001DD3	07A
	TDISK	008200	008200	000800	072
	ADISK	007000	007000	000FFF	062
	TMDCR	008200	008200	000D00	055
	#MDCR	006000	00B818	00035C	051
	AMDCR	007000	007000	001000	041
	RFORMAT	002900	00B818	00013D	03F
	TFORMAT	002300	002300	000200	03D
	RNDTEST	002800	00B818	0000A4	036
	WNDTEST	002800	00B818	0000AA	035
	MDCR1	002900	00B818	000BFA	029
	RAMSTAT	002900	002900	0003FF	04D
	QTH	004000	00C2B2	0009C1	043
	NINPUT	002900	00C2B2	0004CB	03E
	INDEX	002800	00A071	000448	039
	FONEEM	002900	002900	0006FF	032
	ACL-2	008000	008000	001800	01A
	REGFAL	002900	00AFAF	000F30	0C8
	REG.ART	004000	00A071	001BD7	0AC
	6264	002800	007071	0007B2	08C
	MERcode	006000	006FF0	001000	07C
	MERtoel	004000	00A071	00103E	06B
	ACORN	005000	00C2B2	000D1C	05D
	BACKUP4	002900	00C2B2	000B3C	051
	AVE.M	002900	00C2B2	0007F7	049
	MATR.A	004000	00A071	002D47	01B
	MATRIX	002900	00C2B2	0018C0	002

Regioschijf 6-2	SUNDORI	002900	00C2B2	00097B	130
	PARWEER	002900	00C2B2	000676	129
	BUNDESL	002900	00C2B2	000DC9	11B
	GEBODAG	002900	00C2B2	000391	117
	FEESTDA	002900	00C2B2	00062B	110
	TYPWRIT	008200	00C2B2	00093A	106
	BUGBATT	002900	00C2B2	0007EA	0FE
	LEEFTYD	002900	00C2B2	00167B	0E7
	BRAILLE	002900	00C2B2	0011FD	0D5
	BLOBUST	002900	00C2B2	0005FF	0CF
	KIEZEN	002900	00C2B2	00093F	0C5
	ZEKAMER	002900	00C2B2	001413	0B0
	BOWLING	002900	00C2B2	000BD1	0A4
	DOMINO	002900	00C2B2	0010AA	093
	ED64-B	00A000	00A000	001000	083
	SMURF	002900	00C2B2	0020B0	062
	SALFAA	00A000	00A000	001000	052
	DIGITZR	002900	00C2B2	0013CC	03E
	BACKUP	002900	00C2B2	000A39	033
	2440VDU	005000	0056FE	000800	02B
	DUMP	006000	006000	000150	029
	VERBRUI	002900	00C2B2	0026FF	002

De Eprom Programmer aangepast voor 2716. 2764 en 27128.

Uitgangspunt is het schema in A.N.3-1 pag.17 en de software uit A.N.2-6 pag.26.
Om met de 2716 te beginnen. In A.N. heb ik een paar maal gelezen dat de progammer niet werkt voor de 2716. Dat heeft twee oorzaken: in de program-mode wordt de OE doorverbonden met de CE, maar volgens de databladen moet de OE minimaal 2 uS vroeger hoog worden als de CE, daarom heb ik de OE direct aan +5 gelegd. In de read-mode moet de OE laag zijn. De tweede oorzaak zit in de software. De programpuls is negatief, dus gaat hoog-laag-hoog. In afwijking van alle andere eproms moet de programpuls van de 2716 positief zijn dus: laag-hoog-laag. De readpuls is voor alle eproms gelijk en is negatief.
Voor de 2532 werkt de programmer gewoon goed en voor de 2732 hoeven alleen maar een paar aansluitingen worden verwisseld.
De 2764.
Deze eprom is 8k dus er komt een adreslijn bij: A12 en bovendien heeft het IC 28 pennen. De aansluitingen zijn compatible met de 2732, dus 24 pennen. De andere 4 zijn: Vpp, A12, Vcc en PGM. De software aanpassing is eenvoudig: het adresbereik gaat van 4k naar 8k en het startadres van het te copieren programma moet naar het lage geheugen, omdat in het hoge geheugen geen 8k blok meer vrij is. Ik heb gekozen voor #4000 tot #6000.
De 27128.
Deze eprom is 16k dus er komt nog een adreslijn bij: A13. Deze A13 komt aan pen 26(24), waar bij de kleine eproms de +5 aan zit, dus moet dat omgesckakeld kunnen worden. Verder zijn de aansluitingen gelijk aan de 2764. De software aanpassingen zijn: het adresbereik wordt 16k en het startadres van het te copieren programma komt nog lager te liggen, omdat nu een blok van 16k nodig is. Ik heb gekozen voor #3000 tot #7000.
Verder heb ik nog aanpassingen gemaakt voor het programmeren van de A-versies, voor de 2732A moet de Vpp 21V zijn en voor de 2764A en 27128A moet Vpp 12.5V zijn. In het programma worden waarschuwingen gegeven voor het instellen van Vpp.
Fig.1 geeft het schema. De voornaamste wijzigingen zijn de schakelaars. S1a-f verzorgt de omschakelingen voor de diverse eproms, zodat de juiste spanningen op de juiste aansluitingen

staan. S2a-d verzorgt de omschakeling van read naar write. Het voordeel is ook dat de Vpp generator in de read-mode uit staat en geen stroom trekt. S3 tenslotte schakelt de Vpp. De led D1 geeft de program stand aan. De leds D6-D8 geven het epromtype aan. Het omschakelen is tamelijk ingewikkeld geworden, maar is noodzakelijk als de programmer zo universeel moet zijn. Uw eigen wensen zijn natuurlijk aan de hand van het schema aan te passen, zodat het voor specifiek gebruik eenvoudiger wordt.



De schakelaar S2 MOET van het z.g. non-shorting type zijn, anders kost dat een ATOM. S1 hoeft dat niet te zijn als men voor het inprikken van de programmer de schakelaar in de juiste stand zet. S3 is een speciaal type met 3 standen, zodat in de middenstand alle contacten verbroken zijn (on-off-on).

Voor spoel L1 heb ik een geschikt en goed verkrijgbaar type gevonden n.l. 3122 108 9146. uit Philips K9 en K11 KTV. Het is een witte ronde spoel in de voedingsunit en in slooptoestellen ruim voorhanden. De weerstanden R11-R14 moeten uit meerdere weerstanden samengesteld worden om de juiste waarde te verkrijgen. Dit is belangrijk i.v.m. dejuiste progammeerspanning.

De software.

Ik ben uitgegaan van PROM.V2.0. Daarin tot regel 50 de veranderingen aangebracht nodig voor het conditioneel assembleren, de variabelen A,B en C bepalen de read en programpuls. De variabele E het aantal te lezen of te programmeren bytes, de variabele G het begin van de te programmeren bytes, de variabele J bepaalt samen met E hoe hoog de programmeerspanning Vpp moet zijn. Het assembleren gebeurt bij mij standaard op #400, omdat bij de 27128 zoveel geheugen nodig is, ook wordt dan de source overschreven. De variabele D in regel 50 kan dan eventueel gewijzigd worden.

Verder heb ik de regelnummers zoveel mogelijk gelijk gehouden en de aanpassingen tussengevoegd. Veel plezier met de opgevoerde programmer.

Sjaak Geene.
Zonneweide 6,
5221BH 's Bosch.

```
 1 PROGRAM EPROM PROGRAMMER
 3 P.$12;DIMH(0)
 5 P."   *       EPROM PROGRAMMER     *"''
 7 P."   BEFORE STARTING THE PROGRAM"'
 9 P."   SWITCH PROGRAMMER TO THE"'
11 P."   RICHT POSITION"''
13 P."   *    CHOOSE:                 *"'
15 P."   *    FOR        27128 :  1   *"'
17 P."   *    FOR         2764 :  2   *"'
19 P."   *    FOR         2716 :  3   *"'
21 P."   *    FOR         2532 :  4   *"'
22 P."   *    FOR         2732 :  5   *"''
23 DO INPUT"         YOUR CHOISE "F.'
24 UNTIL F>=1 AND F<=5
25 DO XIF F=1 OR F=2 OR F=5 INPUT"          A-TYPE? (Y/N)"$H
26 UNTIL $H="Y"OR $H="N"
27 ELSE
28 IF F=1:A=#EC:B=#CC;C=#CE:E=#40;G=#30
29 IF F=2;A=#EC;B=#CC;C=#CE;E=#20;G=#40
31 IF F=3:A=#CC;B=#EC;C=#EE:E=#08;G=#84
33 IF F=4 OR F=5;A=#EC:B=#CC;C=#CE;E=#10;G=#84
37 XIF F=1 OR F=2 OR F=5 AND $H="Y";J=TRUE
39 ELSE J=FALSE
```

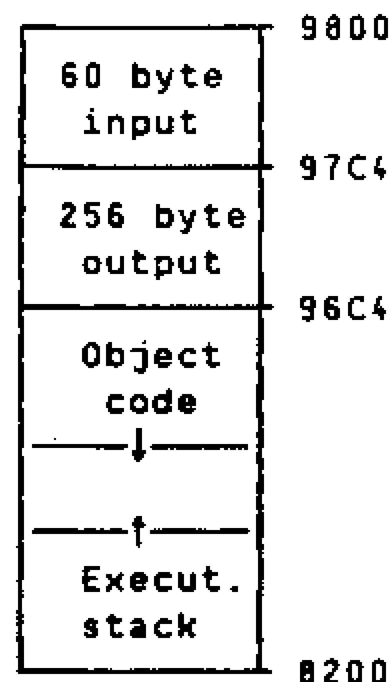
```
 50 P.$12;D=#0400;@=0
 60 P."ASSEMBLING TO #"&D'
 70 DIM VV47; F.N=0 TO 47:VVN=D:N.
 80 F.I=1 TO 2;P.$21;P=D
 90[;\*** INIT ***
100 LDA@#55;STA#208
110 LDA@0;STA#B80E;LDA@#CC:STA#B80C
120:VV7 JSR#F7D1;]
130 ?P=#0C:$(P+1)="EPROM PROCESSOR";!(P+16)=#EA0A0A0D:P=P+20:[
140:VV10 JSRVV0;JSRVV1;JSRVV41;JSRVV34;JSR#CD54;JSR#F7D1;]
150 $P="INSERT EPROM";P=P+12:[NOP
160 JSR#FE94;JSR#CD54
170 \**READ COMMAND**
180:VV11 JSRVV0:JSRVV1:JSRVV41:LDA@#3F:JSR#CD0F
190 LDA#100:CMP@#51:BNEVV33
200 LDA@0:STA#B80C:LDA@#52;STA#208:JMP#C2CF
210:VV33 CMP@#42;BEQVV12;CMP@#52:BEQVV18:CMP@#53:BEQVV26
220 CMP@#57;BEQVV30:CMP@#56:BEQVV21
230 JSRVV6:JSR#CD54:JMPVV11
240:VV30 JMPVV28
250\**BLANK?**
260:VV12 JSRVV0
270:VV13 JSRVV1:JSRVV41:JSRVV5:LDA@#FF:CMP#95:BNEVV14
280 JSRVV2;BEQVV13;JSRVV15;JSRVV16;JMPVV11
290:VV14 JSRVV15;JSRVV6:JSRVV17;JMPVV11
300\**READ**
310:VV18 JSRVV0
320:VV19 JSRVV1:JSRVV41:JSRVV5:LDY@0:LDA#95;STA(#93).Y
330 JSRVV2:BEQVV19:JSRVV20:JMPVV11
340\**CHECKSUM**
350:VV26 JSRVV0
360:VV27 JSRVV1:JSRVV41:JSRVV5
370 CLC;LDA#95;ADC#96:STA#96:LDA@0:ADC#97:STA#97
380 JSRVV2:BEQVV27;JSRVV25:JMPVV11
390\**VERIFY**
400:VV21 JSRVV0
410:VV22 JSRVV1:JSRVV41:JSRVV5:LDA#95
420 LDY@0:CMP(#93),Y:BNEVV23;JSRVV2;BEQVV22
430 JSRVV24:JSRVV16:JMPVV11
440:VV23 JSRVV24:JSRVV6:JSRVV17:JMPVV11
450\**WRITE**
452:VV28 JSRVV0:JSRVV1:JSRVV42:JSRVV31
453 LDA@E:CMP@8:BEQVV36:CMP@#10:BEQVV44:CMP@#20:BEQVV45:JMPVV45
454:VV36 JSRVV39:JSRVV43:JMPVV40
455:VV44 LDA@J:CMP@1:BEQVV37:JMPVV36
456:VV37 JSRVV38:JSRVV43:JMPVV40
457:VV45 LDA@J:CMP@1:BEQVV46:JMPVV37
458:VV46 JSRVV47:JSRVV43:JMPVV40
465:VV40 JSR#FE94;JSR#CD54:JSRVV35:JSR#FE66
470:VV29 JSRVV1;JSRVV42
475 LDY@0:LDA(#93).Y:STA#B800:LDA@A:STA#B80C
480 JSR#FE66:JSR#FE66:JSR#FE66
490 LDA@B:STA#B80C:JSRVV2:BEQVV29:JSR#CD54:JSRVV20
500 JSRVV34:JSR#FE94:JSR#CD54:JMPVV11
510\**INIT ADRESS**
```

```
520:VV0 LDA@#FF;STA#B803
530 LDA@0:STA#91:STA#92:STA#93:STA#96:STA#97
540 LDA@G;STA#94;JSR#FE6B:RTS
550\**WRITE ADRESS**
560:VV1 LDA@#FF;STA#B802:LDA#91:STA#B801:LDA#92:STA#B800;RTS
570:VV41 LDA@#CE:STA#B80C:LDA@#CC:STA#B80C:RTS
575:VV42 LDA@C;STA#B80C:LDA@B;STA#B80C:RTS
580\**INC ADRESS**
590:VV2 INC#93;BNEVV3:INC#94
600:VV3 INC#91;BNEVV4;INC#92
610:VV4 LDA@E:AND#92:RTS
620\**READ BYTE**
630:VV5 LDA@0:STA#B802;LDA@#EC:STA#B80C
640 LDA#B800;STA#95:LDA@#CC:STA#B80C;RTS
650\** ERROR **
660:VV6 JSR#F7D1:]:?P=7:$P+1="ERROR ":P=P+7:[:NOP;RTS
670\**"BLANK CHECK"**
680:VV15 JSR#F7D1:];$P="BLANK CHECK ":P=P+12:[;NOP;RTS
690\**"OK"**
700:VV16 JSR#F7D1:]:$P="OK":P?2=13:P?3=10:P=P+4:[NOP:RTS
710\**"ON #(ADRESS)"**
720:VV17 JSR#F7D1:];$P="ON #":P=P+4:[;NOP
730 LDA#92:JSR#F802:LDA#91:JSR#F802:JSR#CD54;RTS
740\**"VERIFY"**
750:VV24 JSR#F7D1:]:$P="VERIFY ":P=P+7:[:NOP:RTS
760\**"READY"**
770:VV20 JSR#F7D1:]:$P="READY":P?5=13:P?6=10:P=P+7:[NOP:RTS
780\**"SUM=#"**
790:VV25 JSR#F7D1:]:$P="SUM=# ";P=P+5:[;NOP
800 LDA#97;JSR#F802:LDA#96:JSR#F802:JSR#CD54;RTS
810\**"SET program"**
820:VV31 JSR#F7D1:]:$P="SET program":P?11=7:P=P+12
830[;NOP:RTS
840\**"SET read"**
850:VV34 JSR#F7D1:]:$P="SET read":P?8=7;P=P+9
860[;NOP:RTS
870\**"PROGRAMMING"**
880:VV35 JSR#F7D1:];$P="PROGRAMMING":P?11=10;P?12=13:P=P+13
883[:NOP:RTS
885:VV47 JSR#F7D1:]:$P=" 12.5 ":P=P+6;[;NOP:RTS
886:VV38 JSR#F7D1:];$P=" 21 ";P=P+4:[:NOP;RTS
887:VV39 JSR#F7D1;];$P=" 25 ";P=P+4;[;NOP:RTS
888:VV43 JSR#F7D1:]:$P="VOLTS!!!":P?8=10;P?9=13:P=P+10
889[;NOP;RTS
890];N.;@=4
900 P.$6"CODE AT #"&D'"FINISH AT #"&P'
910 P."LENGTH="P-D" (#"&P-D") BYTES"'
915 P."PROGRAMMING FROM #"&G*256
917 P." TO #"&(G+E)*256-1'
920 P."TO START: PRESS KEY"'
930 LINK#FFE3;@=8:LINKD;E.
```

ATOM PASCAL (VER. 1-1986)

LINK#3000 PROMPT: ←

PROGRAM: #3000 - #7FFF
WORKING AREA: #6 - #FF, #200 -#2FF
USER AREA: #8200 - #97FF:



TEXT

```
←T     Top line position
←B     Bottom line position (".")
←N/n   Next n lines (".."=all,RTN=1)
←U/n   Up n lines
←F     Find line containing string
?string
```

IN/OUT

```
←R     Read text from keyboard
?
←L/n   List n lines TV/printer
←C     COS/DOS (To ATOM COS/DOS)
       SAVE"FILE" ---- *LOAD"FILE"
       Back to PASCAL = Control P (Useless after BREAK)
```

TOGGLE

```
←A     Assignment tracing. Print values changed by
       Assignment and FOR statements.
←S     Statement tracing. Print text unit just
       before it is executed.
```

EDITING

```
←D/n   Delete n lines
←I     Insert one line ahead of current line
?
←SP    (SP=SPACE) List current line
←V     View 5 lines
```

EXECUTION

```
←G     GO, execute program
←K     Check program syntax
←,     Execute and trace program
←X     Execute immediately one statement (STM)
?STM

←P     Page, redefine number of printer colums (desimal). RTN=64(default)
←M     Show number of free user memory bytes (desimal)
←Z     Delete program (cold start). N or SPACE = NO,Y = YES
←E     End PASCAL, back to BASIC. (BREAK)
←ESC   Stop execution
```

ATOM PASCAL is both a compiler and an interpreter. It compiles source statements written in Pascal directly from the keyboard into an internal format. The internal format contains both the object code to be interpreted upon execution and source code identifiers for formatted printout to support editing at the source code level. Identation levels are set automatically to aid debugging. The object code (program) can be saved and loaded on tape or disk. (Starting address of code in address #97B2/B3). Input lines must be restricted to 60 characters. Machine-code subroutines can be called using SUBR(CALL) or FUNC(CALL). These subroutines must not contain operating system vectors (#200-#21B):

```
PROGRAM BLEEP;
VAR BLEEP=$FD1A:CHAR;
BEGIN
SUBR(BLEEP)    (or WRITE(FUNC(BLEEP)))
END
```

A well known textbook in Pascal is "Pascal User Manual and Report", Springer Verlag (1978) by Jensen and Wirth. Most of these Standard Pascal features are implemented:

Reserved Words

AND	END	NIL	SET
ARRAY	FILE	NOT	THEN
BEGIN	FOR	OF	TO
CASE	FUNCTION	OR	TYPE
CONST	GOTO	OTHERWISE	UNTIL
DIV	IF	PROCEDURE	VAR
DO	IN	PROGRAM	WHILE
DOWNTO	LABEL	RECORD	WITH
ELSE	MOD	REPEAT	

Reserved Symbols

.	,	'	;		
+	-	*	/	:=	:
=	<=	>=	<>	<	>
(	)	(*	*)	[	]

Word	Function	Operand (s)	Result
ABS	Absolute Value	Integer, Real	Same as Operand
ARCTAN	Arc Tangent	Integer, Real	Real
COS	Cosine	Integer, Real	Real
EXP	Exponent	Integer, Real	Real
LN	Natural Logarithm	Integer, Real	Real
ODD		Integer	Boolean
ROUND	Round	Real	Integer
SIN	Sine	Integer, Real	Real
SQR	Square	Integer, Real	Same as Operand
SQRT	Square Root	Integer, Real	Real
TRUNC	Truncate	Real	Integer

Standard Identifiers

Constants:	FALSE	TRUE		
Types:	BOOLEAN	CHAR	INTEGER	
	REAL	STRING	TEXT	
Functions:	ABS	FUNC	ROUND	SUCC
	ARCTAN	LN	SIN	TRUNC
	CHR	ODD	SQR	
	COS	ORD	SQRT	
	EXP	PRED	SUBR	
Procedures:	READ	READLN *	WRITE	WRITELN
	BREAK			

Character Functions:

Word	Function	Operand (s)	Result
CHR	Character	Integer	Character
ORD	Ordinal	Scalar except Real	Integer
PRED	Predecessor	Scalar except Real	Same as Operand
SUCC	Successor	Scalar except Real	Same as Operand

Machine Language Functions

Word	Function	Operand (s)	Result
SUBR	Subroutine Call	Address	-
FUNC	Function Call	Address, Data	Char

Arithmetic Operators

Symbol/Word	Operation
+	Addition (plus sign)
-	Subtraction (minus sign)
*	Multiplication
/	Division
DIV	Division (yields a truncated integer result)
MOD	Modules (yields the remainder of division)

Relational Operators

Symbol/Word	Operation
>	Greater than
>=	Greater than or equal to
<	Less than
<=	Less than or equal to
=	Equal to
<>	Not equal to

EXECUTION-TIME DIAGNOSTICS

The form of the diagnostic message is

```
ERROR #nn
text unit
```

After control returns to the command interpreter, a series of WRITELNs may be executed with the ←X command to examine the state of the data. The position of the text unit pointer insures that visibility of identifiers is the same as that which prevailed at the time of the error. The data structures built during execution are not initialized until the next ←G or ←, is executed, so data are available for examination.

The "S ERROR" diagnostic indicates a syntax error.

In the case of the syntax error message, the right-most "s" usually gives a strong clue about the error, since the character immediately to its right is the one which led the syntax analyzer into the impasse. If the right-hand "s" is the last character of the diagnostic message, something is missing at the end of the input, for example, a final semicolon in a declaration or definition.

Note: Error numbers with an asterisk are internal checks. Their occurrence may indicate loss of memory integrity.

01*	
02*	
03	Value Stack has exceeded available space.
04*	
05*	
06	Too few actual parameters
07	Too many actual parameters
08	Actual parameter of a VAR formal parameter is an expression.
09*	
10*	
11	Nonordinal type where ordinal (scalar) type required.
12	Maximum permissible dynamic statement depth of 24 exceeded.
13	Ordinal value computation out of range (Array and set selection)
14*	
15*	
16	Expression result type should be Boolean and is not.
17	Function identifier on left-hand side of assignment does not refer to a declared function.
18*	
19	Type compatibility error in assignment or actual value parameter.
20	Type (s) improper for operator.
21	Attempt to make array selection on a nonarray.
22	String length excessive for operation.
23	Empty string improper for operation.
24*	
25	Attempt to negate a nonnumeric.
26	Operand should be Boolean.
27*	
28	GOTO with destination textual depth greater than that of GOTO.
29	Actual/formal parameter type discrepancy.
30	Control variable has changed in FOR loop.
31	Argument of CHR not in 0..255.
32	Value less than lower limit of subrange.
33	Value greater than upper limit of subrange.
34	Argument not ordinal (scalar).
35	Argument not real or integer.

36 GOTO refers to an undefined label.
37 Actual/formal VAR parameter type discrepancy.
38 Array reference to STRING with noninterger subscript.
39 Right operand of IN not a set or packed set.
40 Source String to long for receiving string.
41 Improper argument of record select (.) operator.
42*
43 No match between value of expression and CASE constants.
44 Argument of WITH is not a record variable.
45 More than the maximum of 16 records in a program.
46 Field identifier not preceded by "." and not in the scope of a WITH.
47*
48 Processor stack pointer is higher than when break discontinued execution; Break-in-progress is now false and ↑G will start at beginning.
49 Processor stack is about to overflow.
50 Attempted division by zero.
51 Floating-point overflow.
52 Fixed-point overflow in TRUNC or ROUND.
53 MOD with negative divisor.

Diagnostic messages which occur during binding have the following distinctive form: an indicator line describing the error, followed by one or more program text lines describing where the error occurred. The following indicator lines can occur:

UNDEF identifier
DUP identifier
UNDECL label
DUP label
TYPE identifier

The indicator lines *DUP* identifier and *UNDEF* identifier are each followed by one text unit which contains the cited identifier occurrence.

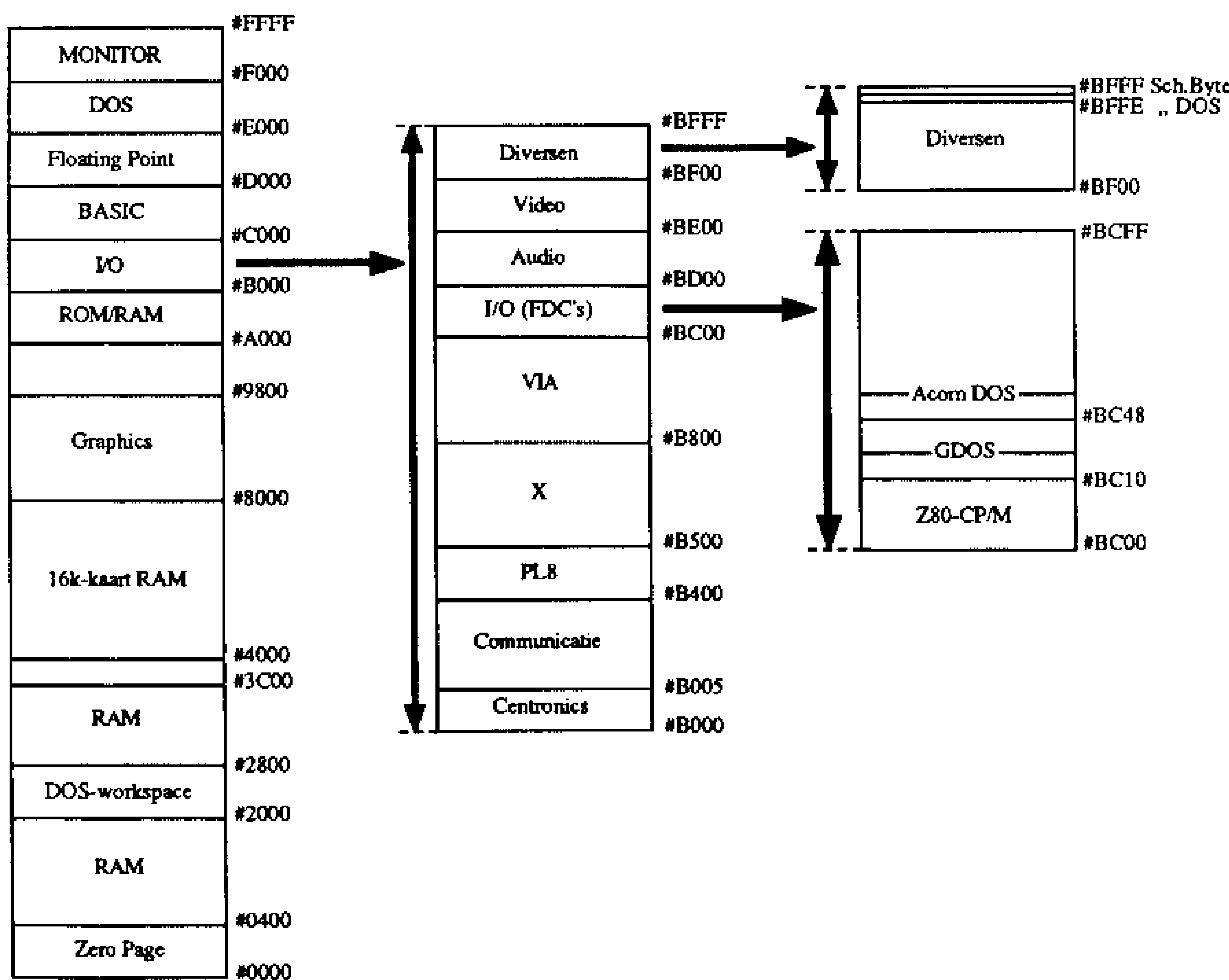
The *DUP* diagnostic is generated at a defining occurrence of an identifier if the identifier has a prior defining occurrence at the same block level.

The *UNDEF* diagnostic is generated at a referring occurrence of an identifier if the identifier has no prior defining occurrence at the same or at any higher block level which would be visible to this referring occurrence.

The indicator line *TYPE* identifier can indicate an inconsistency between the types of the upper and lower bounds of a subrange type; it can also be generated by a STRING type with a noninteger length. The identifier in the indicator line is a type identifier appearing in either the subrange or string type; this identifer refers to a type which is inconsistent with the definition in which it occurs. This definition occurs in the text unit following the indicator line.

Nieuwe adresseringsnormen

Naar aanleiding van het feit dat er weer wat nieuwe clubontwikkelingen plaats nodig hebben in het #BXXX blok hierbij weer een nieuwe memory map met de benodigde uitleg.
De Z80-CP/M kaart heeft een extra VIA nodig welke geplaatst zal worden in het #BXXX blok. De keuze van het definitieve adres is gemaakt op #BC00. Waarom?, wel in het #BCXX blok staan alle diskcontrollers (de GDOS, en de ACORNDOS volgens de nieuwe normwijziging). Het schema om de VIA op het adres #BC00 te krijgen is bijgeleverd bij iedere Z80-CP/M kaart. Voor de mensen die niet zo bedreven zijn in het lussen van zo'n printje kan ik melden dat er een printje zal komen om de VIA te plaatsen. Er zal zo hard mogelijk gewerkt worden om dit in een respectabele tijd voor elkaar te krijgen.
Verder worden er behalve de Z80-CP/M kaart nog wat zaken ontwikkeld zoals een "echte" Centronics interface (ontwikkeling van de regio limburg). De plaatsing hiervoor valt op de plaats van de originele ATOM-VIA namenlijk #B800. Natuurlijk wordt een en ander wel goed uitgedecodeerd. (op 16 bytes nauwkeurig........)
Over de verder in de memory-map besproken zaken kunnen we nog geen goede uitleg geven, maar deze zaken komen wel binnen afzienbare tijd, dus vandaar dat de adressen alvast gereserveerd zijn binnen de I/O-memory-map.
Nogmaals de inmiddels bekende kreet: U bent niet verplicht zich aan de plaatsingsnormen te houden, maar het is wel verstandig, want hoe meer uitbreidingen er in het #BXXX blok komen, hoe meer U uit eigen initiatief geplaatste uitbreidingen in de weg zult zien zitten. Daarvoor adviseert de hardwarecommissie nogmaals: houd alstublieft de clubnormen aan, anders word het voor U een puinhoop in het #BXXX blok.....



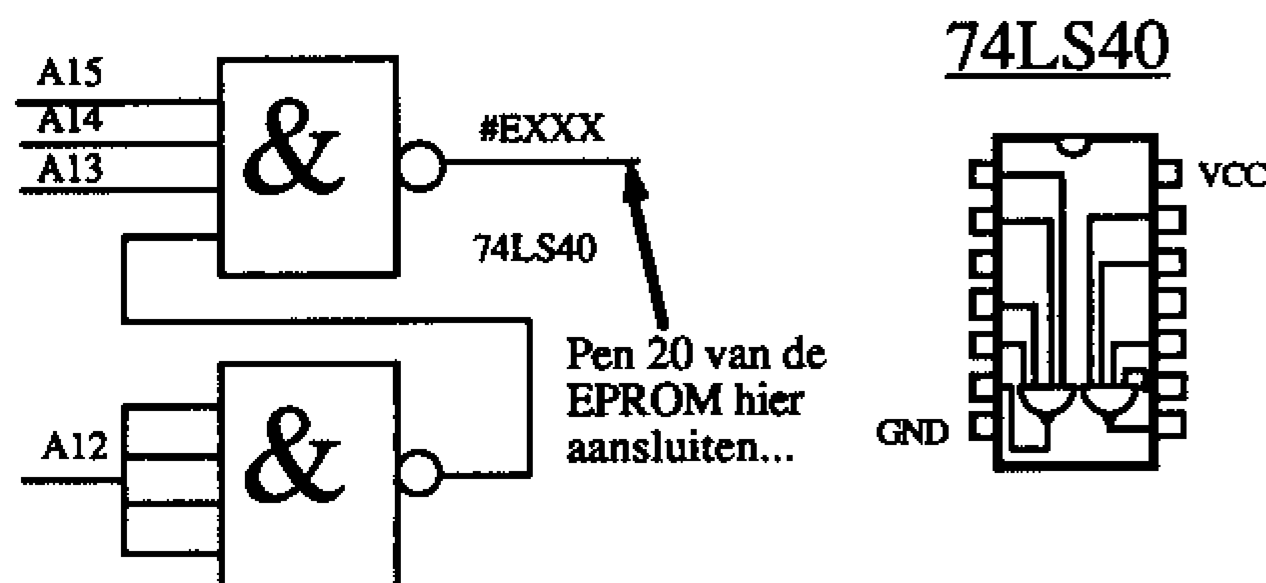
Tips voor de Z80-CP/M gebruikers

De gebruikers van de Z80-CP/M kaart zullen reeds gemerkt hebben dat er bepaalde zaken besproken worden in de handleiding die eigenlijk niet direct mogelijk zijn, omdat er bijvoorbeeld geen ACORNDOS-FDC aanwezig is in het systeem (de EPROM die geplaatst moet worden in het #EXXX blok) In dit verhaaltje zullen we een en ander duidelijk maken over de nodige aanpassingen die aan de ATOM gepleegd moeten worden om de zaak op de Z80-CP/M kaart te laten draaien.
Allereerst de #EXXX EPROM.
Indien U op de schakelkaart geen ruimte meer hebt om een #EXXX blok te plaatsen staan hieronder twee methoden om zonder al te veel problemen een #EXXX blok in de ATOM te creeren.
Methode 1: U heeft een schakelkaart (#AXXX), en geen extra monitorROM op voet 24. (Voet 24 op het ATOM-moederboard is dus geheel vrij.) Maak het decoderingsschakelingetje zoals dat beschreven is in figuur 1, en schrap op het moederbord bij IC24 of voetje 24 pen 20 door, en soldeer een draadje met het selectiesignaal op die plaats. (Het is eventueel ook mogelijk om het pootje van de EPROM uit te buigen, en dan het draadje vast solderen aan dat pootje. Het #EXXX blok is nu voor U beschikbaar.....
Methode 2: U heeft voetje 24 in de ATOM bezet door een EPROM, en deze plaats dient bezet te blijven door diezelfde EPROM. Maak het decoderingsschakelingetje zoals dat beschreven is in figuur 1. Soldeer op de EPROM die geplaatst is in voet 24 een IC-voetje met gedraaide contacten, waarbij de middenverbinding (zie figuur 2) is verwijderd. (Anders zou de ondergelegen EPROM nooit meer gewist kunnen worden.) (Het verdient trouwens aanbeveling om op die onderste EPROM een stikkertje te plakken wat zeer makkelijk verwijderbaar is, omdat het niet echt makkelijk gaat door de opgesoldeerde voet.) Plaats nu op de IC-voet de nieuwe #EXXX DOS-EPROM, en buig het selectiepootje (nr. 20) nu uit. Soldeer op dit pootje de draad die uit de decodering komt, en ziedaar: ook een #EXXX blok beschikbaar.
Andere zaken voor wat betreft de Z80-CP/M kaart:
De VIA...Een probleem op zich want er is op dit moment geen echte VIA-kaart beschikbaar voor de Z80-CP/M kaart. De normplaatsing is zoals al eerder beschreven #BC00. Deze plaatsing kan heel makkelijk, en totaal heeft U slechts drie IC's en 2 connectors nodig voor deze uitbreiding. Let echter wel op het plaatsen van de doorverbindingen aan de VIAzijde, en zorg ervoor dat de connector naar de Z80-CP/M kaart goed vast zit. Voordat U het systeem aanzet is het verstandig om even te controleren of de connector niet verkeerd om zit. De min baan aan beide zijden van de connector moet wel kloppen!
Het schema voor de plaatsing van deze VIA krijgt iedereen die een Z80-CP/M kaart besteld heeft bij de bouwhandleiding. Het schema wat in de grote handleiding staat is inmiddels achterhaald omdat het niet op de plaats #BC00 staat.
Succes met het nabouwen, en veel plezier met de DOS EPROM.

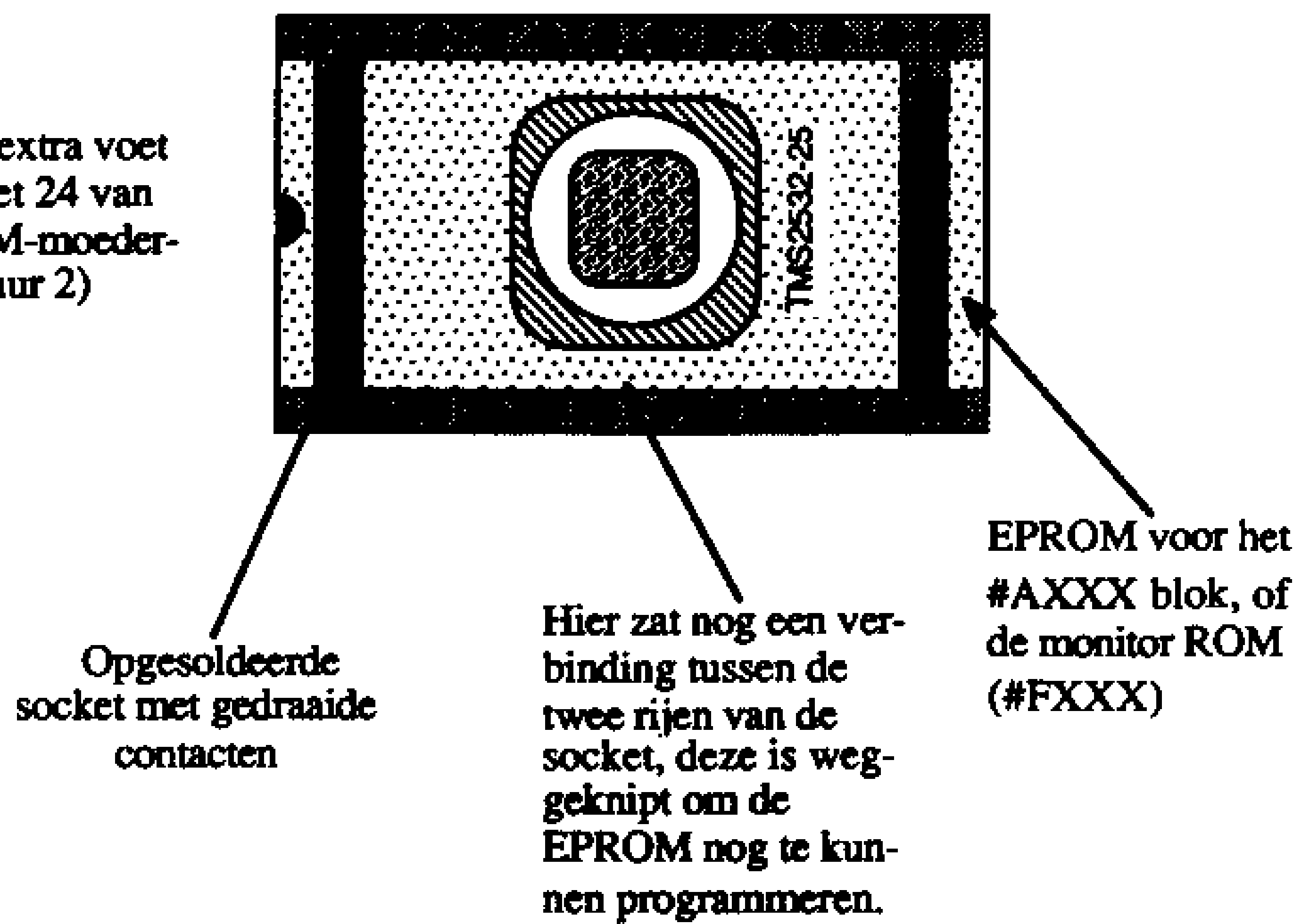
Namens de hardwarecommissie: Yvo Tuk.

Selectie #Exxx Eprom

Figuur 1



U kunt in het ATOM schema kijken op welk IC deze 74LS40 het best geplaatst kan worden als "piggy pack", zodat U de adreslijnen bij de hand hebt.



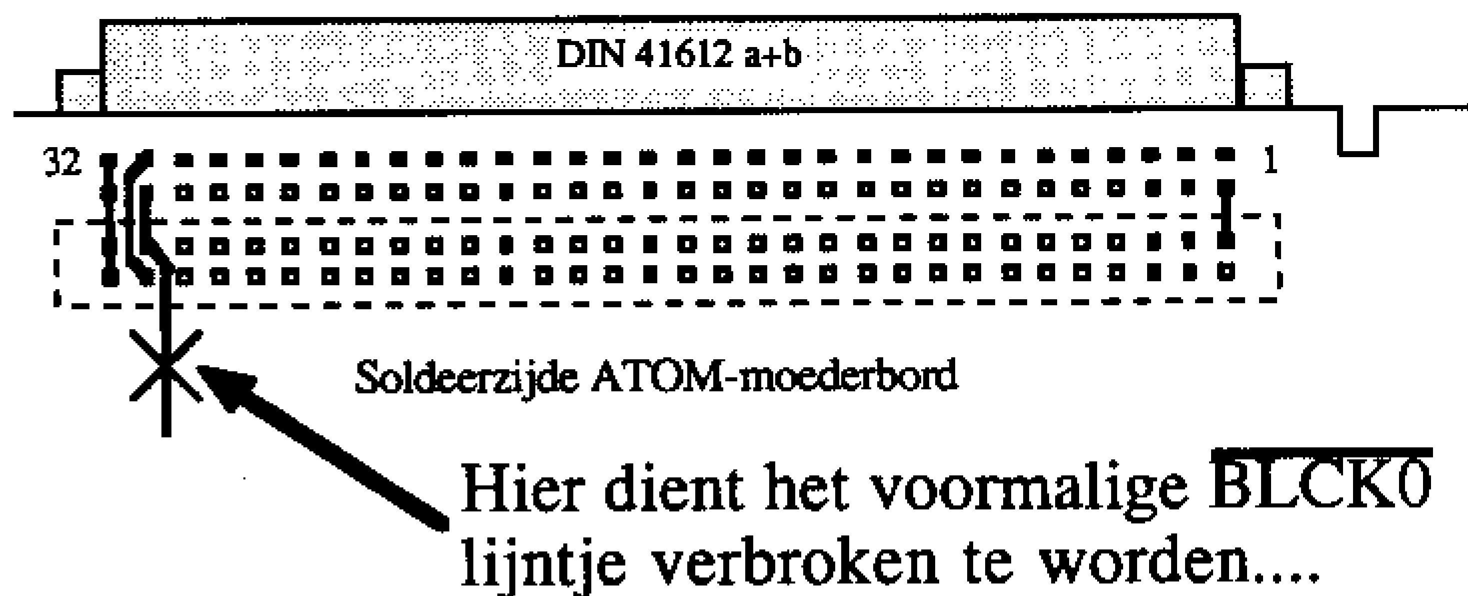
Verplaatsing van de FDC naar #BC48

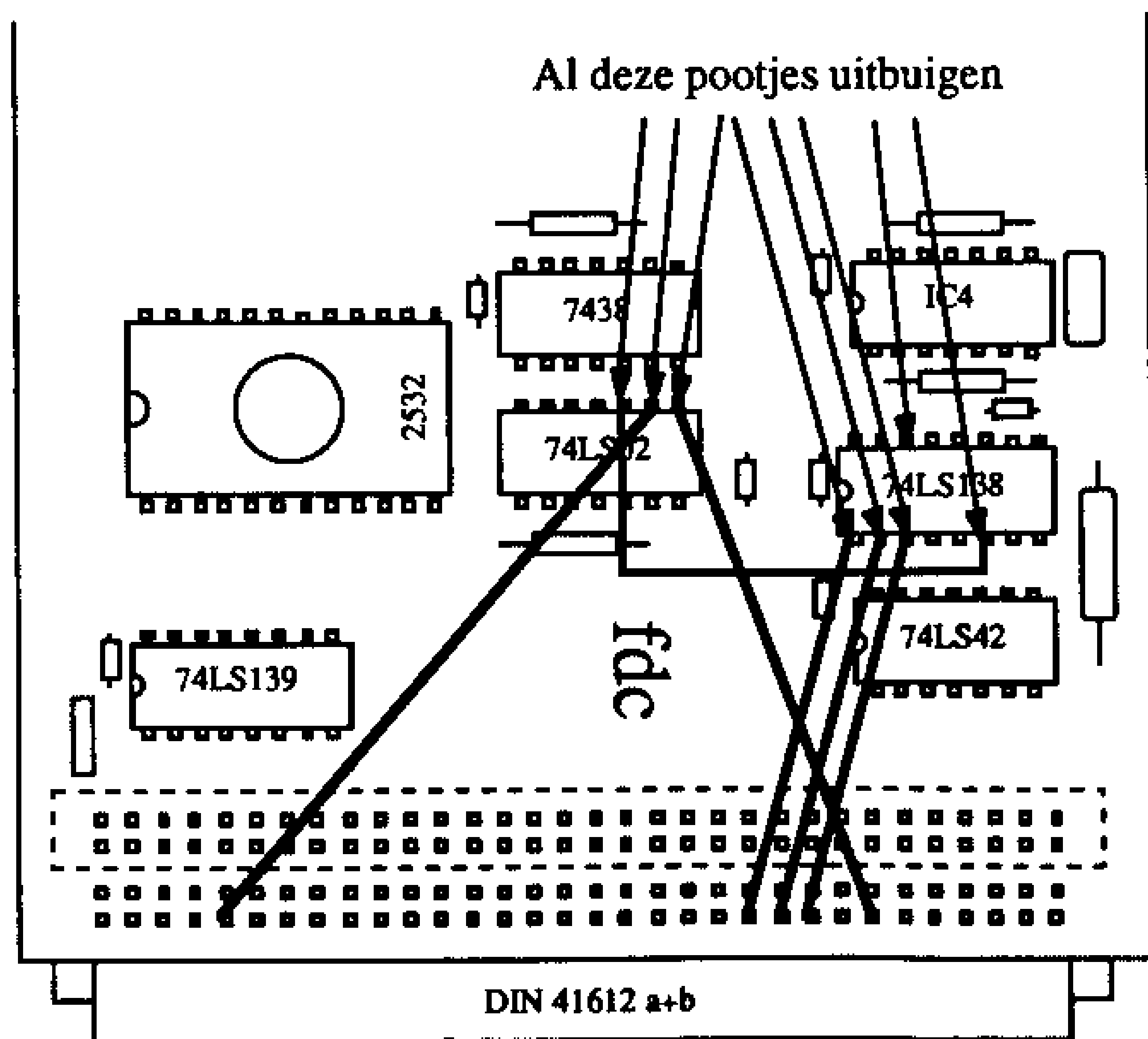
Naar aanleiding van het feit dat het artikel over de verplaatsing van de FDC volgens een nieuwe norm van de Hardwarecommissie op vreemde wijze is verdwenen hierbij nogmaals een artikel over deze verplaatsing.
De verplaatsing komt eigenlijk neer op de manier zoals Nico Stad deze besproken heeft in een voorgaand AN. In deze verplaatsing zat echter nog een klein foutje, want de FDC stond plotseling in ieder blok na het blok #BCXX. Dus nog een opvolgende wijziging......
De methode is vrij simpel; er dient zowel in de Atom als in de FDC wat gewijzigd te worden. Deze wijzigingen zijn te zien in bijgaande figuren. Een korte bespreking van deze figuren:
Figuur 1 & Figuur 2: Het wijzigen van de FDC, aan de componentenzijde, en aan de soldeerzijde.
Figuur 3 & Figuur 4: Het wijzigen van het Atom moederbord aan de componenten- en soldeerzijde.
De motivatie voor de verplaatsing van deze FDC mag wel duidelijk zijn lijkt mij, het hele I/O gebeuren hoort nu eenmaal thuis in het #BXXX blok, en de geijkte plaats hiervoor is het blokje I/O (FDC's). De exacte plaats #BC48 werd gekozen om practische redenen.
De Hardwarecommissie is het dus duidelijk niet eens met het plaatsen van de FDC naar het #EXXX blok; in dat blok hoort de software thuis. Iemand met een uitgebreide DOS zou op het moment dat zo'n wijziging een norm zou worden niet meer mee kunnen met de normen...
De DOS EPROM dient overigens ook gewijzigd te worden, het patch programma is te vinden op de clubschijf die bij het nummer van AN hoorde waar de wijziging van Nico Stad instond...
Succes met het wijzigen!

Namens de hardwarecommissie: Yvo Tuk

Wijziging FDC naar #BC48-#BC4F

Figuur 3



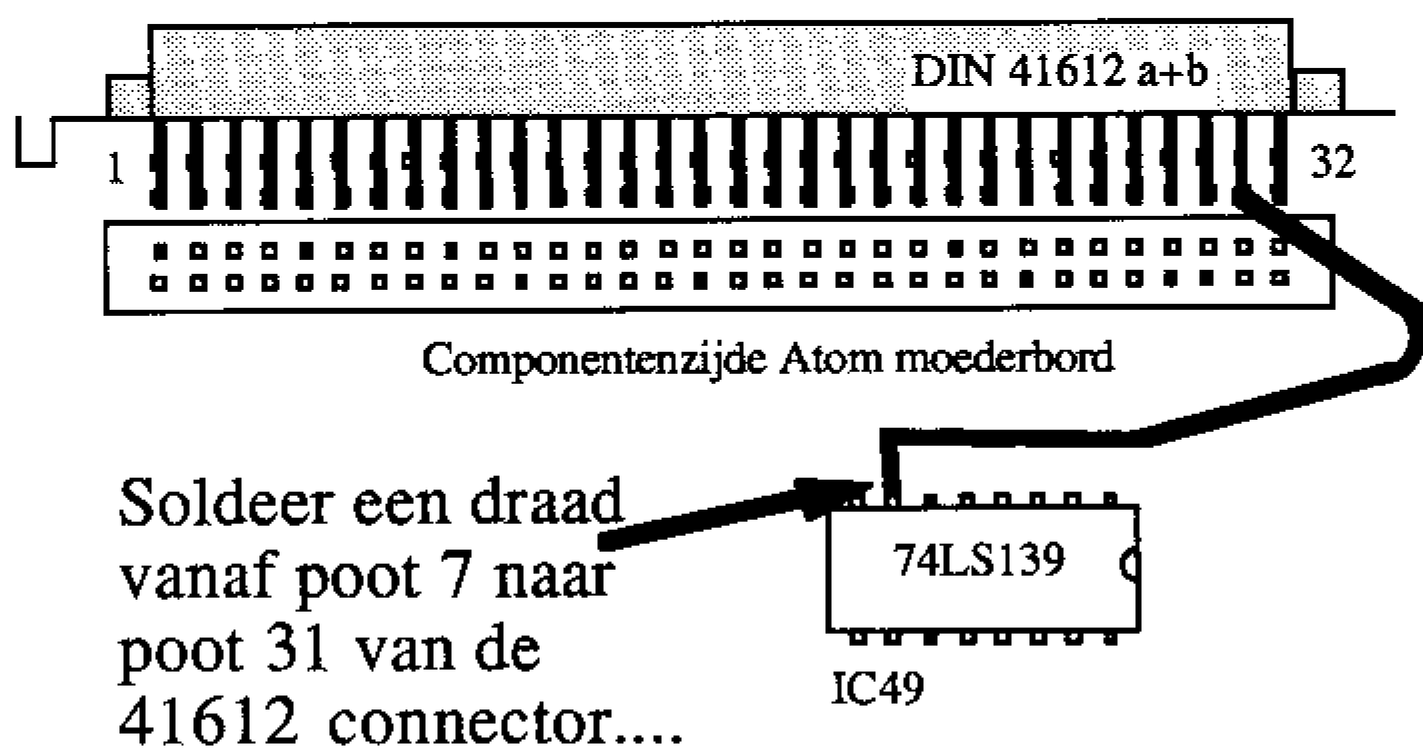


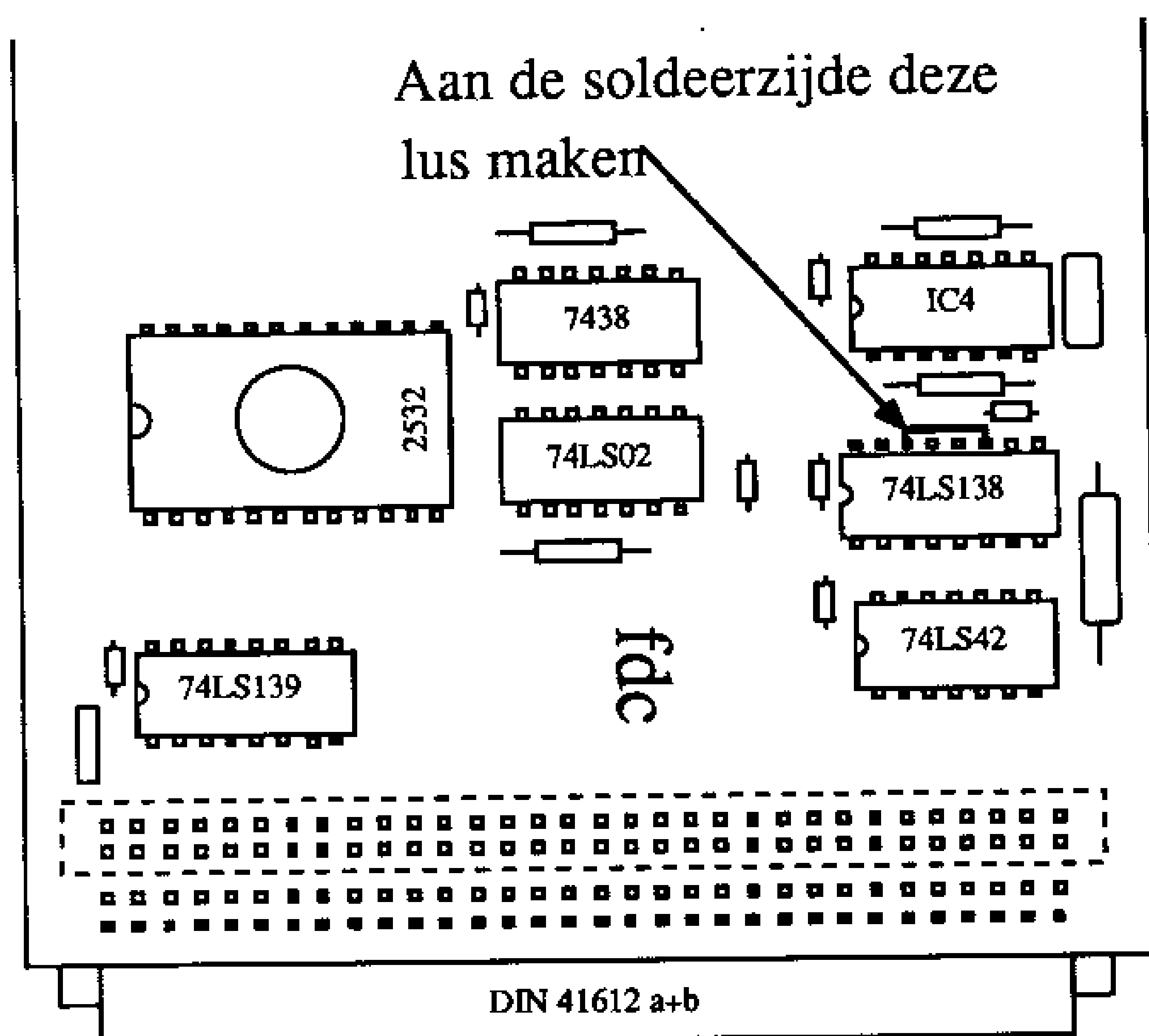
Wijziging FDC naar #BC48-#BC4F

Figuur 1

Wijziging FDC naar #BC48-#BC4F

Figuur 4





Wijziging FDC naar #BC48-#BC4F

Figuur 2

Over MIDI

In reactie op het artikel over MIDI van Theo Waaijer in Atom Nieuws, jaargang 6, nummer 1, en de daarin gestelde vragen doe ik hierbij een poging enkele feiten over MIDI op papier te zetten, en tevens iets van mijn eigen belangstelling voor de combinatie computer+muziek over te brengen. Ik hoop hiermee diegenen die niet met het begrip MIDI vertrouwd zijn zover bij te scholen dat ze eventuele toekomstige verhandelingen of discussies hierover kunnen volgen. Helaas heb ik zelf geen ervaring met het gebruik van MIDI of apparatuur die van een MIDI-aansluiting voorzien is, dus de volgende informatie is uit publikaties afkomstig en niet in de praktijk geverifieerd.

Allereerst iets over de naam MIDI: dit is volgens mijn informatie een afkorting van Musical Instrument Digital Interface, en niet Musical Integrated Digital Interface zoals Theo Waaijer meende te hebben begrepen. Voor de rest van het verhaal is dit natuurlijk van generlei belang.

Wat is MIDI?

MIDI is een gestandaardiseerd interface om elektronische muziekinstrumenten met elkaar te verbinden. Via MIDI kunnen "muziek-data" van het ene instrument naar het andere getransporteerd worden. Waarom zou je zoiets willen? Omdat het zo eenvoudig mogelijk is om bv. met een enkel keyboard meerdere synthesizers tegelijk te bespelen, en bv. muziek die live gespeeld wordt in een sequencer op te slaan en later weer in dezelfde of een andere vorm opnieuw te laten horen. Deze mogelijkheden kwamen pas ter beschikking met de grootschalige invoering van de digitale muziekinstrumenten en de fabrikanten van deze instrumenten besloten (volgens mijn informatie in 1983) tot de invoering van een interface-standaard: MIDI. Daardoor zijn apparaten van verschillende merken, mits voorzien van een MIDI aansluiting, in principe zonder problemen met elkaar te koppelen.

Het MIDI interface transporteert "muziek-data" in principe slechts in een richting, nl. van de MIDI zender naar de MIDI ontvanger. Bidirectioneel datatransport is via MIDI alleen mogelijk door tussen twee apparaten ook twee MIDI verbindingen te maken. Dit is in sommige gevallen wenselijk maar in de meeste gevallen nauwelijks zinvol.
Zoals ik zojuist al opmerkte is er bij MIDI sprake van een zender en een ontvanger. De zender zendt informatie, bv. ingevoerd via een keyboard (een "orgel-manuaal"), in de door MIDI voorgeschreven vorm naar de aangesloten ontvanger(s). Deze informatie vertelt de ontvanger(s) bijvoorbeeld dat er zojuist op het keyboard een G werd aangeslagen en een D werd

losgelaten. Ook kan op deze wijze een ander geluid geselecteerd worden (nieuwe programmakeuze).
Elke ontvanger ontvangt de MIDI informatie, zendt die (eventueel) door naar de volgende ontvanger, selecteert de voor hem bestemde informatie en voert de via MIDI ontvangen instructie uit. Dit kan bijvoorbeeld inhouden dat er een G ten gehore wordt gebracht en het klinken van een D wordt beeindigd.
Er is dus slechts een MIDI zender, die als "Master-keyboard" fungeert, en een of meer MIDI ontvangers die in de vorm van een keten aan de zender gekoppeld zijn, en het eigenlijke geluid produceren. Als Master kan een synthesizer(-keyboard) of een los keyboard worden gebruikt, maar ook een computer.

Het MIDI interface

Het MIDI interface is een asynchroon serieel interface. Daardoor kunnen de verbindingen met eenvoudige DIN kabels gemaakt worden. Het protocol lijkt op dat van het bekende RS232 interface, maar de overdracht-snelheid is hoger, nl. 31250 Baud, om vertraging in de datacommunicatie zoveel mogelijk te beperken. Een byte wordt bij MIDI in de volgende vorm verzonden: 1 startbit, 8 databits, 1 stopbit. Er zijn twee soorten data, te weten statusbytes en databytes. Elk statusbyte wordt gevolgd door een daarbij passend aantal databytes. Een databyte heeft altijd als MSB een 0, een statusbyte als MSB altijd een 1.
Een apparaat dat voorzien is van een MIDI interface kenmerkt zich door de aanwezigheid van 5-polige DIN connectors. Er zijn drie soorten MIDI aansluitingen: MIDI OUT, MIDI IN en MIDI THRU. Een zender verstuurt zijn gegevens via MIDI OUT, een ontvanger ontvangt gegevens via MIDI IN en stuurt ze eventueel door via MIDI THRU. Niet alle apparaten hebben al deze aansluitingen; welke aansluitingen aanwezig zijn hangt af van de mogelijkheden die het apparaat biedt. Een Expander (=synthesizer zonder toetsenbord) heeft bv. geen MIDI OUT want hij kan niet als zender functioneren.

Hoe werkt MIDI?

Welke gegevens worden eigenlijk via MIDI verstuurd? Dat zijn onder andere gegevens over:
- toonhoogte, toonduur
- aanslagdynamiek
- programmakeuze
- pitch bend (toonhoogte modulatie)
- kanaalkeuze (16 verschillende kanalen aanwezig)
- etc.

MIDI kent 16 verschillende kanalen, die onafhankelijk van elkaar kunnen worden gebruikt. Het is hierdoor mogelijk maximaal 16 verschillende apparaten via een MIDI keten onafhankelijk van elkaar te bedienen. Hiervoor bestaan verschillende Modes:

OMNI-Mode: Alle ontvangers negeren de kanaalinformatie en reageren op alle "muziek-data". De zender kent alle "muziek-data" toe aan kanaal 1. Er is in deze Mode dus eigenlijk maar 1 kanaal beschikbaar, en alle apparaten reageren op dezelfde informatie.

POLY-Mode: Alle ontvangers zijn ingesteld op een eigen kanaal en reageren alleen op data die via dat kanaal worden verstuurd. De zender zendt zijn informatie voor elke ontvanger op een apart kanaal. Iedere ontvanger is dus apart aanspreekbaar.

MONO-Mode: Elke stem in een ontvanger krijgt een eigen kanaal toebedeeld en reageert alleen op dat kanaal. Het is dus mogelijk verschillende geluiden uit een instrument tegelijkertijd en onafhankelijk van elkaar te gebruiken. Slechts enkele apparaten ondersteunen deze Mode.

Toepassingsmogelijkheden

Gewapend met deze kennis over MIDI kunnen we ons een beeld vormen over de toepassingsmogelijkheden van een combinatie van MIDI en de Atom. Het meest voor de hand liggend is het gebruik van de Atom als sequencer of als multi-track recorder. Ik zal proberen uit te leggen wat hiermee wordt bedoeld.

Wanneer je als musicus een muziekstuk bedoeld voor meerdere musici in je eentje wilt spelen kom je natuurlijk handen tekort. Dit probleem kun je oplossen door de verschillende partijen na elkaar te spelen en op te nemen, bijvoorbeeld met een bandrecorder. Wanneer je echter elektronische instrumenten met MIDI interfaces gebruikt kun je de verschillende partijen ook via MIDI opnemen en wel in een computer, die daarvoor van een geschikt programma voorzien moet zijn. Deze combinatie noemt men ook wel een sequencer of een multi-track recorder, afhankelijk van de wijze van bedienen. Je kunt namelijk de muziek niet slechts live (in real-time dus) spelen en opnemen, maar ook stap voor stap via keyboard of computer-toetsenbord invoeren, en zelfs naderhand wijzigen.
Een dergelijk computerprogramma zou bv. moeten voorzien in 16 kanalen die onafhankelijk beluisterd en geprogrammeerd kunnen worden, zowel in real-time als stap voor stap, en ieder aan een eigen MIDI kanaal zijn te koppelen. Het dient ook mogelijk te zijn de weergavesnelheid te varieren. Het zou verder nuttig zijn als het programma MIDI informatie in notenschrift zou kunnen omzetten (en omgekeerd) zodat een muziekstuk via een printer in notenschrift op papier kan worden gezet.
De precieze eisen waaraan zo'n programma dient te voldoen zijn natuurlijk niet zomaar even snel op te schrijven, maar moeten zorgvuldig overwogen worden. Overigens bestaan dergelijke programma's al voor enkele home- en personal

computers, o.a. voor de Commodore 64. Aan de technische haalbaarheid (voldoende geheugen, snel genoeg?) hoeft dus niet te worden getwijfeld. Het schrijven van zo'n programma zal echter waarschijnlijk geen peuleschil zijn en heel wat man- en/of vrouw-uren vergen.

Conclusie

Voor mensen die met elektronische muziekinstrumenten werken of dit van plan zijn is MIDI een schitterende uitvinding. De meeste apparaten op dit gebied zijn dan ook standaard van een MIDI interface voorzien. Helaas zijn deze apparaten meestal ook (schrikbarend) duur (enkele duizenden guldens), en dus voor hobbyisten minder interessant.
Daar staat echter tegenover dat het bouwen van een MIDI interface voor de Atom betrekkelijk goedkoop kan blijven, zeker als het gecombineerd kan worden met een universeel serieel interface (RS232, RS423). Op dit gebied zijn er dus geen grote beperkingen.
Of MIDI voor de (gemiddelde) Atom gebruiker van enig nut kan zijn waag ik te betwijfelen. Het eindoordeel hierover laat ik echter graag over aan U, geachte lezer.

Hans van der Linden
Ketelstraat 10
6562 LH Groesbeek
tel. 08891-1336

ZEESLAG TEGEN DE COMPUTER

Al minstens anderhalf jaar geleden ben ik op het idee gekomen om een programma te schrijven waardoor het lijkt of de computer even intelligent (of juist nog intelligenter) reageert als de mens. Een schaakprogramma is hiervan een goed voorbeeld. Omdat ik dat echter veel te ingewikkeld vond, ben ik naar een eenvoudiger vorm gaan zoeken. Dit werd het spel zeeslag.

Het leuke van het programma was dat de intelligentie steeds kon worden uitgebreidt. De eerste programma versie was zeer simpel van opzet. De computer verloor door het slechte spel ruimschoots alle partijen. Ik ben toen uit gaan zoeken met wat voor simpele tacktieken het programma de schepen zo snel mogelijk kon opsporen. Er onstonden zo vele programma versies, met steeds betere tacktieken. Nu is het is zo dat in principe de computer even goed is als de speler (geluk speelt echter ook een rol).

Het programma is geheel opgebouwd uit procedures en functies. Het gemis van echte variabelennamen i.p.v. de letters A t/m Z werd als zeer storend ervaren. Om het programma te kunnen "runnen" heeft men het volgende nodig:
- een "standaard" atom
- RAM van #2800 tot #9800 (sorry, het programma is vrij lang)
- P-charme

Het geheugen wordt als volgt ingedeeld:
- #2900 tot #5B00 programma zeeslag
- #5B00 tot #61A0 extra P-charme statements (GRMOD, POKE en SPRITES)
- #61A0 tot #6200 assemblerprogramma voor verwisselen van het grafische-scherm
- #6200 tot #6300 sprites tabel
- #6300 tot #6800 arrays en werkruimte P-charme (t.b.v. o.a. de procedures en functies)
- #6800 tot #8000 beeldscherm buffer

Om een goede grafische weergave te verkrijgen wou ik gebruik maken van sprites, dat zijn figuurtjes die je met 1 commando kunt tekenen. Ik heb hiervoor een simpel programma gemaakt in de vorm van het statement SPRITES. Met het statement POKE kan er op een eenvoudiger manier dan met '!' of '?' een sprite in het geheugen worden opgeslagen. Een voorbeeld:

```
POKE #6100,1,8,#7E,#83,#03,#7E,#C0,#C0,#C1,#7E;REM letter 'S'
```

Hier worden 10 bytes vanaf adres #6100 achter elkaar in het geheugen opgeslagen. Dit is een sprite met als startadres #6100 waarmee de letter 'S' kan worden getekend. De eerste 2 bytes geven de lengte en de hoogte van de sprite aan. Deze sprite is 1 byte, dus 8 "puntjes" lang, en tevens 8 "puntjes" hoog. Bij een geset bit wordt er een puntje geinverteerd (wit wordt zwart en omgekeerd) en bij een gereset bit wordt er niets gedaan. De sprite kan als volgt worden uitgeschreven:

```
#7E      .XXXXXX.
#C1      XX.....X
#C0      XX......
#C0      XX......
#7E      .XXXXXX.
#03      ......XX
#83      X.....XX
#7E      .XXXXXX.
```

Om een eenvoudige definitie van een sprite te waarborgen, moet een sprite altijd een lengte van een of meerdere bytes in de X-richting hebben. Een byte omvat 8 bits of "puntjes". Bij langere sprites wordt de opbouw als volgt:

```
POKE #6100,2,3,#FF,#F8,#08,#08,#FF,#F8;REM rechthoek - 13*3
```

betekend:

```
#FF #F8    XXXXXXXXXXXXX...
#80 #08    X...........X...
#FF #FF    XXXXXXXXXXXXX...
```

Dit is dus een sprite van 13*3. Met het commando:

```
MOVE 10,10;SPRITES #6100
```

wordt de sprite bij coordinaat 10,10 geplaatst. Met deze 2 statements kunnen dus eenvoudige sprites worden gemaakt. De GAGS-ROM biedt natuurlijk veel meer mogelijkheden, maar zorgt echter voor storingen als men daarnaast tegelijkertijd intensief gebruikt maakt van P-charme. De statements POKE en SPRITES zijn ook als losse P-charme statements meegegeven.

Bediening

Bij zeeslag moeten er een aantal schepen worden opgespoord en tot zinken gebracht. De volgende schepen zijn aanwezig:

```
- 1 slagkruiser, 4 "hokjes" lang en mag niet aan de wal liggen
- 1 kruiser      4    "       "   "    "    "   "  "   "    "
- 3 jagers       3    "       "   " mogen "   "  "   "    "
- 3 torpedoboten 2    "       "   "   "  wel  "  "   "    "
```

Alleen de kleine torpedoboten mogen aan de wal liggen (dit zijn de hokjes 0,0 t/m 0,9, 9,0 t/m 9,9, 0,0 t/m 9,0 en 0,9 t/m 9,9).

Deze informatie wordt na het opstarten geprint. Daarna wordt er gevraagd of men zelf een speelbord voor de computer wil opzetten, of dat de computer dit voor je moet doen. Wanneer men zelf een bord wil opzetten moeten achter elkaar de coordinaten worden ingetypt van de verschillende schepen. Het programma controleert of dit correct gebeurt. Het programma plaatst zijn schepen "at random". Soms komt het voor dat de schepen ongelukkig worden geplaatst, waardoor er bijna geen ruimte meer is voor alle schepen. Wanneer dit voorkomt zal er met een schoon speelbord opnieuw worden begonnen.

Nadat de speelborden gereed zijn, doet de computer het eerste schot. Daarna komt het lege speelbord van de speler te voorschijn. Er wordt gevraagd:

COORD:

Men typt hierop eerst de X-coordinaat en daarna de Y-coordinaat in. Als men zich hierin vergist, dan typt men i.p.v. van de tweede coordinaat een <DEL> in, waarop er opnieuw wordt begonnen. Het resultaat, een T voor een "treffer" of een X voor een

"misser" (plons), wordt nu gegeven. Na een treffer moet men het hele schip tot zinken zien te brengen. Rondom een tot zinken gebracht schip worden door de computer "puntjes" gezet. Dit om aan te duiden dat er op deze plaatsen geen ander schepen kunnen liggen (de schepen mogen elkaar niet raken). Deze "puntjes" heten geheugensteuntjes. Men kan zelf ook geheugensteuntjes plaatsen aan het einde van ieder beurt. Wil men dat niet dan geeft men een willekeurige toets.

De speler moet winnen van de computer. Dus bij een gelijk aantal zetten wint de computer. Ik hoop dat jullie veel plezier beleven aan dit spel en dat het vooral aanspoort tot zelf programmeren. Dat is veel leuker dan alleen het "runnen" van andermans programma's.

Klaas Ypma
Parkweg 140
3134 VS Vlaardingen
tel.: 010-4346517

Dit is het volgende deel (2) in de serie gestruktureerd programmeren. Om de draad weer op te nemen, nemen we de tekeningen: Tek.6, Tek.7 en Tek.8 uit de vorige aflevering weer ter hand.

In de PSD's hebben we dus ook 3 elemtaire loop's die prachtig aansluiten bij onze loop's in BASIC nl.:

De Voor <loopteller variable> := <beginwaarde> TOT-EN-MET <eindwaarde>
 = FOR <variabele> = <beginwaarde> TO <eindwaarde>

De "TOTDAT testwaarde 'waar' wordt" loop.
 = DO .... --> UNTIL (testwaarde = true)

De "ZOLANG testwaarde 'waar' is " loop.
 = WHILE (testwaarde= true) --> WEND

Nu even een voorbeeldje voor het gebruik van de PSD's. B.v. voor bestands manipulatie: Het inlezen en wegschrijven van een serieel bestand van en naar cassette.

Elk bestand heeft een markeerpunt dat aan geeft: hier is het einde van het bestand. Dit kan een door het programma gegenereerd eindpunt zijn, bv. een record met als inhoud: 'EINDE' of bv. '-9999'. Het kan ook een punt zijn dat door het operating system is geplaatst, het EOF (end of file) byte. Voor het gemak nemen we de eerste mogelijkheid, en noemen we dit het EOF-teken. Nu is het zo dat we het EOF-teken niet mogen verwerken, daar het geen data is dat door het programma verwerkt moet worden.Als we dus een bestand gaan lezen, lezen we totdat we het EOF-teken zien, en stoppen dan met het inlezen. Er geldt nu dus:

1: lees eerste record:
2: vergelijk met EOF-teken, en stop als gelijk:
3: verwerk de data, en lees volgende record:
4: ga naar punt 2:

Hieronder zien we van de gevraagde programma's de PDS's weergegeven, met de namen LEESBEST en MAAKBEST

Tek. 9 en Tek. 9a

Leesbest

initialiseer en open bestand als input
lees EERSTE record uit Best.
zolang record<> '-9999'
verwerk rezord
lees volgende record

Maakbest

initialiseer en open bestand als output
Druk AF "stoppen = '-9999'"
voer in instring
Schrijf instring naar Best.
TOTDAT instring = '-9999'

Hieronder staan de programma's in BASIC uitgewerkt.

```
10 PROGRAM LEESBEST
20 DIM B(30)
30 A=FIN""
40 SGET A,B
50 WHILE NOT($B="-9999")
60   PRINT $B';REM verwerk data
70   SGET A,B
80 WEND
90 END
```

```
10 PROGRAM MAAKBEST
20 DIM A(30)
30 B=FOUT""
40 PRINT"stoppen= '-9999'"'
50 DO
60   INPUT $A
70   SPUT B,$A
80 UNTIL $A="-9999"
90 END
```

We kunnen met PSD's meerletterige variabele-namen gebruiken, met zinnige namen, zodat het nog overzichtelijker wordt. Voor ATOM's die dit toestaan (het gebruik van meer letterige variabele-namen) is het dus nog eenvoudiger om van een PSD een programma te maken.

Hoe kunnen we nu een idee voor een programma omzetten naar een PSD?. Allereerst: we vergeten dat we een computer hebben met daarop een of andere programmeertaal. Een PSD kan dus naar elke beschikbare programmeertaal worden omgezet. Pas als de PSD klaar is, en het werkt op papier (ontdekt door te testen), kunnen we hem overzetten naar een bepaalde programmeertaal.

Het testen van een programma op papier, noemen de programmeurs 'droogzwemmen', ze liggen dan vaak languit over de vloer tussen de tekeningen.Dit testen gaat als volgd: ze nemen aan aantal input waarden in gedachten, en lopen dan de werking van het programma na tot ze uitkomsten verkrijgen, en deze uitkomsten worden dan vergeleken met reeds bekende uitkomsten, het programma is pas goed als de uitkomsten overeen stemmen.

Laten we als voorbeeld een sorteerprogramma gaan ontwerpen, dat een tabel met 10 elementen sorteerd in oplopende volgorde. We verwerken dit idee weer in een PSD en daarna, als het werkt, in BASIC.

We beginnen met het opzetten van een globale PSD, waarin is verwerkt wat er grofweg gebeuren moet, hoe het precies moet gebeuren, is niet belangrijk, dit schuiven we voor ons uit (doen we later in PROCEDURE's of subroutine (stap voor stap ontwikkelen)). Wat moet het programma doen?, in de eerste plaats hebben we een stuk initialisatie (het klaar zetten van de data e.d. voor het sorteer prog.). Als tweede hebben we de sorteer routine zelf, en als derde een routine die de gesorteerde data afdrukt..

Hieronder staat het globale PSD weergegeven, we noemen dit het 'Sorteer'-programma.

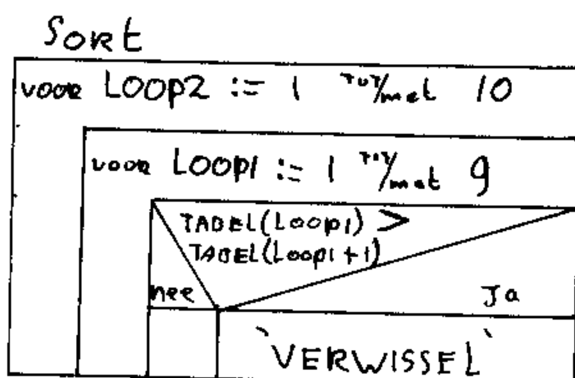
Tek.10 **Sorteer**

'INITIALISEER'
'SORT'
'DRUK AF TABEL'

Nu beginnen we met de sorteer routine zelf: Hoe sorteren we?: we kijken naar de eerste twee elementen in de tabel, als het eerste element groter is dan het tweede, moeten we deze 2 omkeren en dan kijken we naar het tweede en derde element enz. enz.
De volgorde van kijken is dus: 1+2 dan 2+3 dan 3+4 enz. tot en met 9+10. Dit zijn dus 9 mogelijkheden om te wisselen, en dit doen we 10 maal achter elkaar (we hebben 10 elementen in de tabel).

Hieronder staat het PSD van de sorteer routine zelf, we noemen hem 'SORT'.

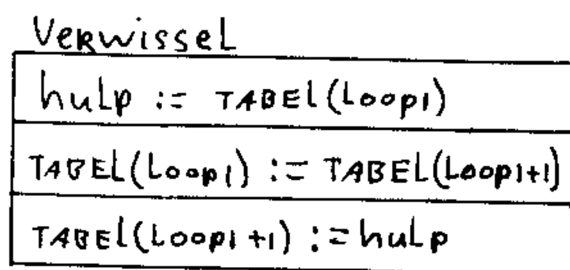
Tek.11



Deze sorteer methode noemen we 'BUBBLE-SORT', de getallen borrelen als het ware naar hun goede plaats. probeer maar op papier met een aantal getallen aan de hand van het programma.

Hier zien we al de mogelijkheden van gestruktureerd programmeren. Bij SORT zien we een loop LOOP1 binnen de loop LOOP2 liggen (genest), en verder zien we dat er een subroutine VERWISSEL wordt aangeroepen vanuit SORT als door de uitkomst van de vraagstelling in het beslissingsdiagram blijkt dat het eerste getal groter is dan het tweede getal.

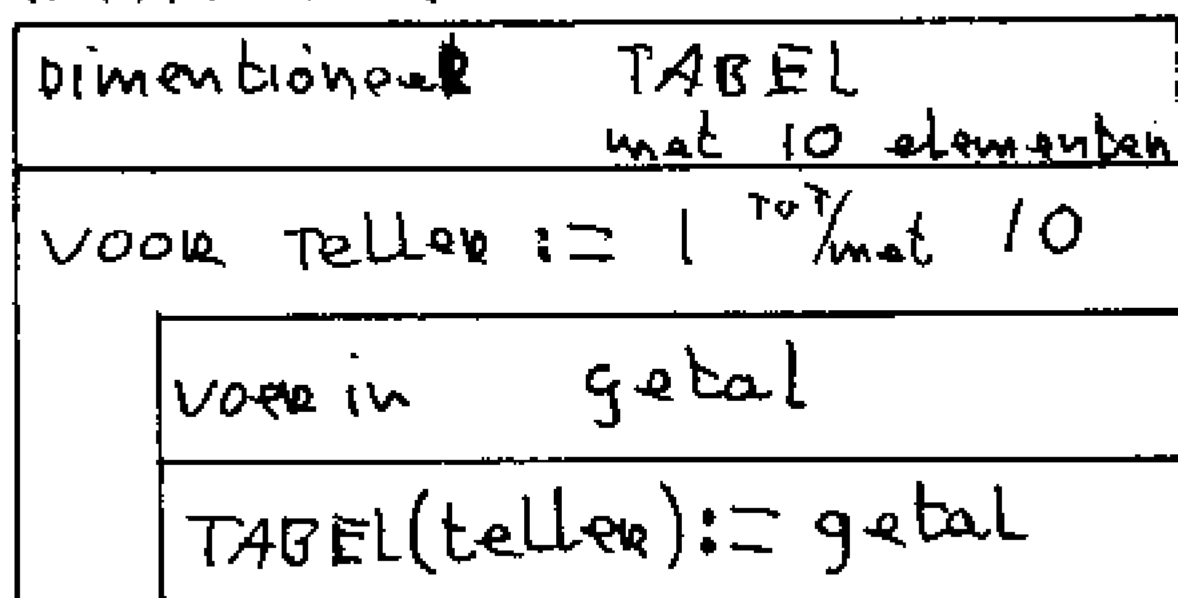
Tek.12



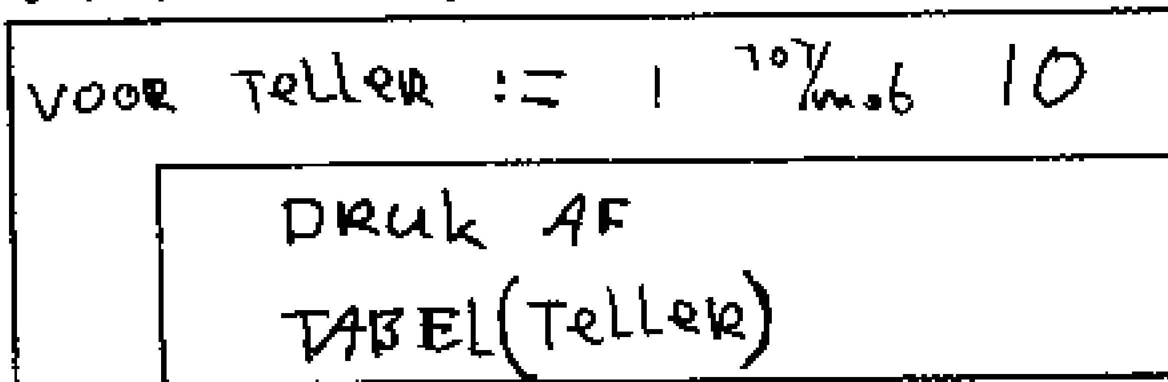
Hiernaast staat de routine VERWISSEL uitgewerkt, we gebruiken een hulp variabele om tijdens het wisselen van de getallen, een getal te onthouden, daar het anders verloren zou gaan. Zoals u al opgemerkt hebt, worden subroutines die aangeroepen worden, in het aanroepende programmadeel tussen "" of '' geplaatst en de naam wordt bij het programma zelf er boven geschreven.

Hieronder staan de overgebleven delen van het sorteer programma (het PSD voor 'INITIALISEER' en 'DRUK AF TABEL'). Zoals is te zien, is de routine SORT ook afzonderlijk te gebruiken, omdat de hieronder gegeven routine's alleen de invoer en uitvoer voor de SORT routine verzorgen.

Tek.13



Tek.14



N.B. dimentioneer niet binnen procedure of functie (zie P-CHARM)

Tot zover dit deel. Probeer alvast aan de hand van deze PSD's een BASIC programma te schrijven (M.b.v. LOOP's, PROCEDURES of subroutines). In het volgende clubblad zal ik mijn oplossing geven.

Martie Krukkeland.
Lettele Ov.

Niet geheel tevreden met de reeds eerder gepubliceerde Ramtest-programma's , maakte ik deze zoveelste versie . Ze is bedoeld voor herkenning door P-Charme en heeft de volgende eigenschappen:

- TRANSPARANT , d.w.z. programma's worden niet gewist,
- NOGAL SNEL , 32k wordt in pakweg 4 seconden getest,
- INFORMATIEF , geeft foutieve adressen op 't scherm.

Het is echter niet mogelijk om ram te testen in het gebied van de code zelf , dit lijkt mij vanzelfsprekend.
De regels 50-60-70 dienen ter initialisatie , zoals voor een P-Charme tabel noodzakelijk is , zie ook A.N. 3-2.
Wilt u het RAM-statement in een box plaatsen , dan dient u het af te sluiten met JMP#C558 (regel 240).

syntax: RAM,#xxxx,#yyyy (xxxx= begin , yyyy= eind)

```
10 PROGRAM RAM STATEMENT
20
30 REM jan bronzwaer (ACL)
40 DIMLL4;F.I=0TO4;LLI=-1;N.;P.$12"ram-STATEMENT"''
50 P."TABEL OP: ";IN.T;IN."CODE  OP: "A;P.$21;F.X=0TO1;P=A
60 GOS.a;N.;P.$6';@=0;T!0=#00C6E3FF;T=T+3;$T="RAM";T=T+L.T
70 T?0=LL0/256!#80;T?1=LL0%256;T?2=#80;T=T+2;?#3FC=T/256
80 P."LENGTE CODE: "(P-A-1)'"ZEROPAGE:#90-#93"'';END
90a
100[
110\interpreteer commando
120:LL0 JSR#C78B;JSR#C231;JSR#C4E1;LDA#16;STA#90;LDA#25
130 STA#91;LDA#17;STA#92;INC#92;LDA#26;STA#93;LDY@0;STY#04
140\berg geheugeninhoud op
150:LL1 LDA(#90),Y;STA#94;JSR#C504
160\test adres met #FF
170 LDA@#FF;STA(#90),Y;LDA(#90),Y;CMP@#FF;BNELL3
180\test adres met #00
190 LDA@#00;STA(#90),Y;LDA(#90),Y;CMP@#00;BNELL3
200\zet geheugeninhoud terug
210 LDA#94;STA(#90),Y
220\verhoog teller tot einde
230:LL2 LDX@#90;JSR#FA08;BNELL1;JSR#FFED;JSR#FFED
240 JSR#F7D1;];$P="ready";P=P+L.P;[;NOP;JSR#F7FD;JMP#C55B
250\print foutief adres
260:LL3 LDA#E0;CMP@27;BPLLL4;LDA@35;JSR#FFF4
270 LDA#91;JSR#F802;LDA#90;JSR#F802;JSR#F7FD;JMPLL2
280:LL4JSR#FFED;JMPLL3
290]
300 RETURN
```

Het NINPUT (numerieke INPUT) statement is bedoeld om te worden gebruikt in een door P-Charme te herkennen tabel.

Syntax : NINPUTx , waarin x = maximaal 64.

U krijgt een vraagteken op het scherm en mag vervolgens alleen getallen intypen met een maximale lengte x (dit om bijvoorbeeld een sprong naar de volgende regel te voorkomen) , waarna AUTOMATISCH return volgt. Bij minder dan x karakters moet u zelf een return geven. Hetgeen u intypt wordt als STRING opgeslagen in de stringbuffer op #140. Daar halen we het in Basic op met het statement VAL.

Wat is toegestaan , wat is verboden ?

- Slechts EEN "-" teken en slechts EEN "." zijn toegestaan en wel in de goede volgorde. NINPUT laat in dat geval geen onzin toe en tikt u op de vingers door middel van piepjes en cursor-bevriezing.
- "Deleten" mag ook , de cursor stopt vanzelf zodat de tekst VOOR het vraagteken niet wordt opgegeten.
- E-notatie is verboden , u typt decimaal getallen in.
- Geeft u slechts een return dan staat er alleen #0D in de stringbuffer. Dit levert met VAL de waarde 0 op.
- Wilt u het getal "oneindig" invoeren, dan typt u "o" (= shift O). De tekst "oneindig" verschijnt nu op het scherm en in de stringbuffer wordt "2E30" gedumpt . Met VAL$#140 zitten we dan groot genoeg , dacht ik.
- Met de ESCAPE-toets kunt u uit NINPUT ontsnappen zonder dat uw basicprogramma stopt. In dat geval staat er "^" in de stringbuffer.

Hoe halen we nu de numerieke invoer ons programma binnen? Welnu dat begrijpt u wellicht beter aan de hand van een klein voorbeeldje :

```
10 PROGRAM EXAMPLE
20 DIM A(64)
30 NINPUT6
40 %A=VAL$#140
50 $A=$#140;IF$A="^";GOTO 70
60 FPRINT %A;END
70 P."NOGMAALS"';GOTO30
```

In regel 30 wordt $A erbij gehaald omdat de Atom bij het IF-statement de stringbuffer op #140 gebruikt en diens inhoud dus verandert. Ook moet regel 40 altijd VOOR regel 50 staan. Laat ik niet vergeten Charl de Moor eens openbaar te wijzen op zijn didactische kwaliteiten . Hij maakte op mijn verzoek de eerste opzet voor dit statement en zei vervolgens : "en de rest kun je zelf!". ZO leer je assembler , waarde clubgenoten.

```
 10 PROGRAM NINPUT STATEMENT
 20
 30 REM jan bronzwaer (ACL)
 40 DIMLL10;F.I=0TO10;LLI=-1;N.
 50 P.$12"NINPUT-STATEMENT"'''
 60 IN."TABEL OP: "T
 70 IN."CODE  OP: "A;P.$21
 80 F.X=0TO1;P=A;GOS.a;N.
 90 P.$6';@=0;T!0=#00C6E3FF
100 T=T+3;$T="NINPUT";T=T+L.T
110 T?0=LL0/256|#80;T?1=LL0%256
120 T?2=#80;T=T+2;?#3FC=T/256
130 P."LENGTE CODE: "(P-A-1)'
140 P."ZEROPAGE:#90-#93"'';END
150a
160[
170:LL0
180 JSR#C4E1;LDX@0;STX4
190 LDA#16;STA#90
200 LDA@#3F;JSR#FE52
210 LDY@0;LDX@0;STY#92;JMPLL2
220:LL1
230 STX#91;JSR#FD1A;LDX#91
240:LL2
250 STX#91;JSR#FE94
260 LDX#91;LDY#92
270 CMP@#6F;BEQLL8
280 CMP@#1B;BEQLL9
290 CMP@#7F;BEQLL6
300 CMP@#0D;BEQLL7
310 CPX@#00;BNELL3
320 CMP@CH".";BNELL3
330 LDX@#FF;JMPLL5
340:LL3
350 CPY@#00;BNELL4
360 CMP@CH"-";BEQLL5
370:LL4
380 CMP@#30;BCCLL1
390 CMP@#3A;BCSLL1
400:LL5
410 JSR#FE52;STA#140,Y
420 INY;STY#92
430 CPY#90;BEQLL7;BNELL2
440:LL6
450 CPY@#00;BEQLL1
460 JSR#FE52;DEC#92
470 LDA@CH".";CMP#13F,Y
480 BNELL2;LDX@0;JMPLL2
490:LL7
500 LDA@#0D;STA#140,Y
510 JSR#FFED;JMP#C55B
520:LL8
530 JSR#F7D1;];$P="ONEINDIG"
540 P=P+L.P;[;NOP
550 LDY@#00;LDA@CH"2"
560 JSRLL10;LDA@CH"E"
570 JSRLL10;LDA@CH"3"
580 JSRLL10;LDA@CH"0"
590 JSRLL10;JMPLL7
600:LL9
610 LDA#B001;AND@#20;BEQLL9
620 LDA@#5E;JSRLL10;JMPLL7
630:LL10
640 STA#140,Y;INY;RTS
650]
660 RETURN
```



En zo komen de mooiste programma's tot stand.

Omdat er nogal wat MODEMS in de club zijn en omdat er een VIDITEL programma in de club is wil ik over de kosten van het een en ander toch wel wat vertellen.
Viditel is zoals U weet een informatie- en communicatiesysteem tegen, zoals de PTT zelf zegt, relatief geringe kosten.
Of dat inderdaad zo is kunt U zelf vaststellen aan de hand van de onderstaande gegevens.
Een van de voordelen van dit systeem is dat er een tweerichting verkeer mogelijk is en herbergt nogal wat andere functies in zich.
Zo kunt U bijvoorbeeld bestellingen doen dmv zgn. antwoordbeelden (de electronisch bestelkaart) of U kunt berichten verzenden via
VIDIBUS. Ook kunt U wereldwijd telexberichten verzenden en ontvangen via VIDITEX.
U kunt informatie betrekken uit computers van derden door gebruik te maken van VIDIPOORT.
Al deze toepassingen kunnen door alle VIDITEL-abonnees worden gebruikt.
Men heeft hiervoor nodig:
-- telefoon (dat had u niet gedacht he)
-- VIDITEL-terminal bv een ATOM
-- MODEM

Wat kost dat geintje nu allemaal?
Het aansluiten van een MODEM door de PTT kost U eenmalig fl. 30,-- (excl. BTW)
Het maandelijks-VIDITEL abonnement kost:
-- zonder (PTT) modem fl. 10,-- per maand (excl. BTW)
-- met (PTT) modem fl. 17,50 per maand (excl. BTW)
Het PTT modem kunt U ook kopen, dan kost U dat fl. 292,-- excl. BTW. Let er wel op dat dit modem NAGENOEG ALLEEN voor VIDITEL te gebruiken is.

En wat kos het bellen nu zelf?:

Overdag van 8.00-18.00 uur:	computertijd 10 ct per minuut excl. BTW
	telefoontijd 15 ct per 5 minuten vrij van BTW
's Avonds 18.00-08.00 uur:	computertijd 8,6 ct per minuut excl. BTW
	telefoontijd 15 ct per 5 minuten vrij van BTW

Deze tarieven gelden op 1 januari 1987.

De VIDITELcomputers in Amsterdam en Den Haag zijn bereikbaar via een 06-nummer. Door de invoering van de 06-nummers is het mogelijk alle VIDITEL-abonnees onafhankelijk van hun woonplaats een goedkoop tarief te laten betalen. De VIDITELnummers zijn:
06-8421 voor de computer in Den Haag en
06-8422 voor de computer in Amsterdam

Wilt U nog meer informatie belt U dan even met 0017.
Ik hoop met dit artikel het misverstand dat VIDITEL duur is uit de wereld te hebben geholpen.

Het volgende programma is bedoeld voor gebruikers van de "SP 0256-AL2", waarmee het mogelijk is spraak op te wekken m.b.v. uw computer. Schakelingen hiervoor hebben al diverse malen in -toen nog- "Acorn nieuws" gestaan en in de elektuur van mei '86. Het is wel nodig dat deze aangesloten is op de printeruitgang van uw Atom, zoals ook in het elektuur-artikel is vermeld. Verder moet uw Atom voorzien zijn van P-CHARME. Met dit programma is het mogelijk op eenvoudige wijze woorden te creeren door de reeds ingevoerde fonemen regelmatig te laten uitspreken en dan weer verder in te voeren. Na het geven van "RUN" ziet u het volgende menu op het scherm :

-----------/menu*-----------*

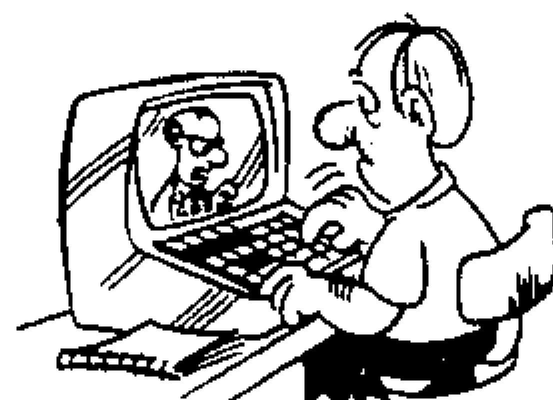
1=INVOEREN

2=UITSPREKEN

3=WIJZIGEN

4=VERDER INVOEREN

5=LIJST FONEMEN



MAAK UW KEUZE?

U kan nu een keuze maken door de gewenste cijfertoets in te drukken, "RETURN" is niet nodig. ALs u voor "5" kiest ziet u alle beschikbare fonemen in alfabetische volgorde. Met "1" kunt u starten met de fonemen-invoer. Nu typt u de fonemen in die u nodig denkt te hebben, steeds gevolgd door "RETURN". Als u stoppen wilt met de invoer typt u nogmaals "RETURN". Met "2" kunt u de fonemen laten uitspreken, op het scherm worden dan alle ingetypte fonemen zichtbaar gemaakt, voorafgegaan door een nummer. Als de Atom een pauze (PA1...PA5) tegenkomt wordt op de volgende regel verdergegaan. Als blijkt dat u ergens een verkeerde foneem gebruikt hebt, kiest u (in het menu) voor "3", waarna u het nummer van de verkeerde foneem intypt. Dan kunt u de juiste foneem intypen, die dan op de plaats van de oude wordt gezet. Met "4" kunt u de lijst verder invoeren. Dan volgt nu het programma :

```
 10 PROGRAM FONEEM
 20 REM W.H.A.SCHOENMAKERS
 30 DATA"PA1","PA2","PA3","PA4","PA5","OY","AY","EH","KK3"
 40 DATA"PP","JH","NN1","IH","TT2","RR1","AX","MM","TT1"
 50 DATA"DH1","IY","EY","DD1","UW1","AO","AA","YY2","AE"
 60 DATA"HH1","BB1","TH","UH","UW2","AW","DD2","GG3","VV"
 70 DATA"GG1","SH","ZH","RR2","FF","KK2","KK1","ZZ","NG"
 80 DATA"LL","WW","XR","WH","YY1","CH","ER1","ER2","OW"
 90 DATA"DH2","SS","NN2","HH2","OR","AR","YR","GG2","EL","BB2"
100 DIM AA(63),I(4),K(4)
110 RESTORE
120 F.X=0TO63;DIM J(10)
130 AA(X)=J;READ$AA(X);N.
```

```
140 S=#5000;REM adres tabel
150 PROC INVOEREN;P.$12
160bIN."TOETS DE FONEEM IN"$I
170 X=0
180cIFX>63 P.$7"INVOER NIET CORRECT!!"';P.$11$11;G.b
190 IF$I=""T.Y=Y-1;G.a
200 IF$AA(X)=$I T.S?Y=X;Y=Y+1;G.b
210 X=X+1;G.c
220aPEND
230 PROC UITSPREKEN
240 P.$12;@=0;F.C=0TOY;G=S?C
250 IFG<5T.P.'
260 P.$21$2$(G+64)$0$3$6
270 P.C"="$AA(G)" ";N.C
280 P.';P."toets";LINK#FFE3
290 PEND
300 PROC WIJZIGEN
310 P.$12;IN."NUMMER VAN FONEEM"B
320dP.';IN."WAT IS DE NIEUWE FONEEM"$K
330 X=0
340fIFX>63 OR $K=""G.d
350 IF$AA(X)=$K T.S?B=X;G.e
360 X=X+1;G.f
370ePEND
380 PROC LIJST
390 P."PA1.PA2.PA3.PA4.PA5.AA.AE.AO.AR.AW.AX.AY.BB1.BB2.CH."
400 P."DD1.DD2.DH1.DH2.EH.EL.ER1.ER2.EY.FF.GG1.GG2.GG3.HH1."
410 P."HH2.IH.IY.JH.KK1.KK2.KK3.LL.MM.NG.NN1.NN2.OR.OW.OY.PP."
420 P."  RR1.RR2.SH.SS.TH.TT1.TT2.UH.UW1.UW2.VV.WH.WW.XR.YY1."
430 P."YY2.YR.ZH.ZZ"
440 P.'"toets";LINK#FFE3
450 PEND
460gP.$12;P."*------------*/menu\*------------*"''
470 P."1=INVOEREN"''
480 P."2=UITSPREKEN"''
490 P."3=WIJZIGEN"''
500 P."4=VERDER INVOEREN"''
510 P."5=LIJST FONEMEN"'''
520 P."MAAK UW KEUZE?";INKEYZ
530 IF$Z=CH"1"Y=0;INVOEREN;G.g
540 IF$Z=CH"2"UITSPREKEN;G.g
550 IF$Z=CH"3"WIJZIGEN;G.g
560 IF$Z=CH"4"P.$12;Y=Y+1;INVOEREN;G.g
570 IF$Z=CH"5"P.$12;LIJST;G.g
580 G.g
590 END
```



Opmerkingen :

In regel 140 moet de variabele "S" het startadres van de tabel bevatten waar de fonemen worden opgeslagen. Als u op #5000 geen RAM-geheugen heeft kunt u hier een ander adres opgeven, bijv. S=#3000.

De fonemen PA1 t/m PA5 zijn de pauzes, ze wekken dus geen klanken op.

Met dit programma kan men het volgende berekenen :
A) Afstandsberekening
B) Coordinaten decimaal
C) Coordinaten in graden, minuten en seconden
D) Eigen antennerichting
E) Antennerichting tegenstation

Er is een Josbox bodig
Met dit programma kunt U ca. 1866240000 verschillende brekningen maken
voor inlichtingen: 04750-33925

PAOCCR en PAOWCR

Dan volgt hier het programma. Ook dit programma is weer via het disk- en het bandjesarchief te verkrijgen.

```
 10 REM QTH AFSTANDSBEREKENING (MAIDENHEAD)
 20 REM VOOR ACORN ATOM BEWERKT DOOR PAOCCR EN PAOWCR ROERMOND
 30 REM VERSIE 010986 --- JOSBOX NODIG
 35 ON ERR P."ANTENNERICHTINGEN ONBEREKENBAAR!"';E.
 40 P.$12
 50 F.W=1TO31;P.$#A3;N.;P.';DOP.$#DF;U.C.=3
 60 P."ww"$128"qth"$128"afstandsberekening"
 70 DOP.$#DF;U.C.=31;P.';F.W=1TO31;P.$#D0;N.;P.''
 75 P."              maidenhead"''''
 80 P." (BEST 73 ES GD DX DE PAOCCR)"'
 85 P."             (ES PAOWCR)"'''''
 90 P."         DRUK SPATIEBALK !!!"
100 LINK #FFE3
110aP.$12
120 DIM N(6),Q(6),P(1)
130 P.$7;IN."MYN QTH-LOCATOR "$N
140 $Q=$N;GOS.1;GOS.k
150 %U=%X;%V=%Y
160 P.$7;IN."ZYN QTH-LOCATOR "$Q
170 GOS.1;GOS.k
180 %Z=%Y-%V
190 %D=C.%Z*C.%U*C.%X+SIN%U*SIN%X
200 %D=-ATN(%D/SQR(-%D*%D+1))+PI/2
210 %O=%D
220 %D=%D*111.2*180/PI
230 D=%(%D+0.5);@=1
240 IF D=0 P."U TIKTE 2 GELYKE QTH-LOCATORS IN"';G.b
242 P.$7$7$7
245 P."AFSTAND IS             : "D" KM."'
250 %C=(SIN(%X)-SIN(%U)*COS(%O))
260 %C=%C/((COS%U)*SIN%O);%C=ACS%C;%C=%C*180/PI
270 C=%(%C+0.5);@=3
274 FIF %V>%Y C=360-C
275 P."MYN ANTENNERICHTING : "C" GRADEN"
277 IF C<=180 P."ZYN ANTENNERICHTING : "C+180" GRADEN"
278 IF C>180 P."ZYN ANTENNERICHTING : "C-180" GRADEN"
280bIN."NOG EEN BEREKENING (J/N)"$P
281 IF $P="J" G.a
282 IF $P="N" P.$12
283 P."               DIT WAS :"''''''''
```

```
284 F.W=1TO31;P.$#A3;N.;P.';DOP.$#DF;U.C.=5
285 P."campers"$128"ham"$128"computing"
286 DOP.$#DF;U.C.=31;P.';F.W=1TO31;P.$#DO;N.;P.';E.
3001REM X-Y INDEX ROUTINE
310 A=Q?0-65
320 IF A>17 OR A<0 G.f
330 B=Q?1-65
340 IF B>17 OR B<0 G.f
350 F=Q?2-48
360 IF F<0 OR F>9 G.f
370 H=Q?3-48
380 IF H<0 OR H>9 G.f
400 R=Q?4-65
405 IF R>23 OR R<0 G.f
415 S=Q?5-65
417 IF S>23 OR S<0 G.f
420 %X=(-90+B*10+H+S/24+1/48)*PI/180
422 FIF %X<0 FP.%X*180/PI" GRADEN Z.B."'
425 FIF %X>=0 FP.%X*180/PI" GRADEN N.B."'
430 %Y=(-180+A*20+F*2+R/12+1/24)*PI/180
435 FIF %Y<0 FP.%Y*180/PI" GRADEN W.L."'
436 FIF %Y>=0 FP.%Y*180/PI" GRADEN O.L."'
440 R.
450fP."NIET BESTAANDE QTH-LOCATOR OF "'
455 P."TIKFOUT !!!!!!!!!!"'
456 DOP.$7;U.C.=10
460 G.b
585 REM GMS OMZET-ROUTINE
590k%E=%X*180/PI
595 @=0
610 E=%E
630 %J=(%E-E)*60
640 J=%J
660 %G=(%J-J)*60
665 G=%G
670 FIF(%G-G)>=.5;%G=%G+1
675 IF %G>=60;%J=%J+1
676 IF %G>=60;%G=%G-60
677 IF %J>=60;%E=%E+1
678 IF %J>=60;%J=%J-60
679 FIF %X>=0 P.%E"."%J"."%G" N.B."'
680 FIF %X<0 P.%E"."%J"."%G" Z.B."'
690 %T=%Y*180/PI
710 T=%T
720 @=0
730 %M=(%T-T)*60
740 M=%M
760 %W=(%M-M)*60
765 W=%W
770 FIF (%W-W)>=.5;%W=%W+1
775 IF %W>=60;%M=%M+1
776 IF %W>=60;%W=%W-60
777 IF %M>=60;%T=%T+1
778 IF %M>=60;%M=%M-60
780 FIF %Y>=0 P.%T"."%M"."%W" O.L."'
790 FIF %Y<0 P.%T"."%M"."%W" W.L."'
800 R.
```



<<<<<<<<<<<<>>>>>>>>>>>>
<<<< H I G H W A Y >>>>
<<<<<<<<<<<<>>>>>>>>>>>>

Highway is een spelletje voor de GAGS V2.3 en maakt af en toe gebruik van Josbox of P-Charme om het beeldscherm te kopieren. Als in Uw Atom geen schakelsoft aanwezig is kunt U dit vervangen door een FOR/NEXT -loop. Het programma loopt in een Atom zonder gestapeld geheugen, verder is ook een joystick nodig (indien U geen joystick bezit zie de opmerking na de listing).

LINKS A B C D E RECHTS

Het spelen: Als het spel begint staat Uw auto (DAF 66) op de E-positie. Het is de bedoeling om naar de A-positie te rijden om daar een liftster op te halen en haar naar de overkant (de E-positie) te brengen. Ontwijk hierbij de hindernissen ! Als U erin slaagt om drie liftsters naar de overkant te brengen rijdt Uw auto het Benzine-Station binnen. Hier kunt U punten winnen door op de oliepomp drie dezelfde tekens te plaatsen. Dit kunt U proberen door op de vuurknop te drukken. Het spel eindigt als U drie maal tegen een hindernis aanbotst.

```
   10 REM ***** highway *****
   20 REM ** ROLAND  LEURS **
   30 T=0;GAGS;BASE#86
   40 CREATE/M CAR1,56,56,56,144,168,147,65,127,0,0,0,1,1,129
,2,254/A:101
   50 CREATE/H CAR2,56,56,56,144,168,147,65,127,56,56,56,17,41
,145,2,254/A:102
   60 CREATE/H CAR3,102,255,189,255,66,60,0,0,0,0,0,0,0,0,0,0
   70 FOR A=5TO6
   80 CREATE/H OBS3,31,255,63,255,6,2,1,0,255,248,254,248,4,6
,0,0/A:A;N.
   90 FOR A=1TO2
  100 CREATE/V OBS1,216,72,124,58,57,56,56,84,124,16,56,56,56
,68,0,0/A:A;N.
  110 FOR A=3TO4
  120 CREATE/V OBS2,0,0,24,24,24,24,24,24,24,62,231,195,0,0,0
,0/A:A;N.
  130 CREATE/V HITC,0,54,36,36,36,127,62,28,60,92,62,9,28,28,8
,0
  140 CREATE/V BUSS,0,0,24,24,24,24,24,24,24,255,130,67,254,0,
0,0
  150 CREATE/H CRAS,0,0,12,19,36,9,2,0,0,0,12,50,209,72,16,0
  160 CREATE/V TREE,60,24,24,28,26,24,24,56,124,252,254,126,60
,56,0,0
  170 CLEAR2;MOVE14,0;DRAW24,96
  180 MOVE 113,0;DRAW103,96
  190 SET:TREE,3,85;IF ABSRND%2=1 THEN TURN:TREE
  200 SET:TREE,1,19;IF ABSRND%2=1 THEN TURN:TREE
  210 SET:TREE,7,55;IF ABSRND%2=1 THEN TURN:TREE
  220 SET:TREE,120,90;IF ABSRND%2=1 THEN TURN:TREE
  230 SET:TREE,121,32;IF ABSRND%2=1 THEN TURN:TREE
```

```
240 SET:TREE,118,63;IF ABSRND%2=1 THEN TURN:TREE
250 FOR B=24 TO 103;MOVE B,0;DRAW B,96;NEXT B
260 FOR X=15 TO 23;Y=0
270 PIXEL X,Y,Z;IF Z=0 THEN PLOT 13,X,Y
280 Y=Y+1;IF Z<>0 THEN NEXT X
290 IF X=24 THEN GOTO 310
300 GOTO 270
310 FOR X=104 TO 112;Y=0
320 PIXEL X,Y,Z;IF Z=0 THEN PLOT 13,X,Y
330 Y=Y+1;IF Z<>0 THEN NEXT X
340 IF X=113 THEN GOTO 360
350 GOTO 320
360 INV;COPY #8000 #8600 #9000
370 P. $12;?#E1=0;P."********************************"
380 P. "*********** highway ***********"
390 P. "*******************************"
400 P. '"LINKS   A   B   C  D   E  RECHTS"''
410 P. "HET SPEL BEGINT - DE AUTO BEGINT"
420 P."BIJ  DE E POSITIE - VERMIJDT  DE"
430 P."HINDERNISSEN ALS DE  AUTO  OP DE"
440 P."B,C OF D POSITIE STAAT - NEEM 'N"
450 P."LIFTER MEE - PROBEER OP DE E POS"
460 P."TE KOMEN MET LIFTER."''''"          <TOETS>";INV;LINK#FFE3
470  P.$12;?#E1=0;P."score:"''"<1> ONTWIJK EEN HINDERNIS  : 10
P"
480 P."<2> NEEM EEN LIFTER MEE :   20 P"
490 P."<3> ZET DE LIFTER AF :      50 P"
500 P."<4> JACKPOT (DRUK TOETS)";INV;LINK#FFE3
510 P.$12;?#E1=0;P."NADAT DRIE LIFTERS ZIJN  AFGEZET"
520 P."KAN DE  AUTO HET  BENZINESTATION"
530 P."BINNEN RIJDEN. PROBEER  DEZELFDE"
540 P."DRIE NUMMERS OP  DE OLIEPOMP  TE"
550 P."KRIJGEN  DOOR OP  DE VUURKNOP TE"
560 P."DRUKKEN."''"111    --->   50 P"'
570 P."333    --->  100 P"'"777   --->  300 P"'
580 P."===    --->  500 P"'"ANDER --->    0 P"''
590 P."<DRUK TOETS>";INV
600 LI.#FFE3;CLEAR2;COPY#9000 #9600 #8000
610 G=101;SETG,93,10;L=0;H=0;K=0;U=0;P=0;D=0;M=20
620 SET1,36,1;SET2,36,96
630 SET3,55,60;SET4,55,-45
640 SET5,74,32;SET6,74,128
650 DO A=0;B=0;C=0;JOYSTK A,B,C
660 POSG,D,E
670 IF A=1 GOSUBr
680 IF A=255 GOSUBl
690 GOSUB o;PAUSE M
700 UNTIL P=3;GOTOe
710r IFD=74;IFH=0ORK=0;R.
720 IFD=93;R.
730 UNSETG;SETG,(D+19),10;SOUND50,30;R.
740l IFD=36;IFL=0ORK=1;R.
750 IFD=17;R.
760 UNSETG;SETG,(D-19),10;SOUND50,30;R.
770oIFL=0;IFA.R.%15=9;SET:HITC,7,96;L=1
780 IFL=1;SHOVE:HITC,0,-8;POS:HITC,D,E;IFE<3;UNSET:HITC;L=0
790 IFE=16ORE=8;IFK=0;POSG,D,E;IFD=17;UNSETG;G=102;SETG,17,
10;K=1;L=0;UNSET:HITC;U=U+20;GOS.p
```

```
 800 IFH=0;IFA.R.%15=2;SET:BUSS,114,96;H=1
 810 IFH=1;SHOVE:BUSS,0,-8;POS:BUSS,D,E;IFE<3;UNSET:BUSS;H=0
 820 IFE=16ORE=8;IFK;POSG,D,E;IFD=93;UNSETG;G=101;SETG,93,10
;K=0;H=0;UNSET:BUSS;U=U+50;GOS.p;O=O+1;IFO=3;G.j
 830 F.A=1TO6;SHOVEA,0,-8;ATHIT A,G;GOSUBc
 840 POSA,D,E;IFE>0A.E<9;POSG,D,E;IF D>17A.D<93;U=U+10
 850 NEXT;SOUND10,30;R.
 860cPOSA,D,E;UNSETA;GOS.s;F.Q=0TO7
 870 SET:CRAS,D,E;PAUSE7;NEXTQ;P=P+1;IMAGE:CAR3,(P*10-6),96
 880 UNSETG;G=101;SETG,93,10;SETA,D,E;K=0;R.
 890eF.Z=1TO2;PAUSE10;GOS.s;N.
 900 P.$12"================================="
 910 P. "========== HIGHWAY ==========="
 920 P."================================="'';@=0
 930 FILL#8060 #8200 64;INV
 940 IFU>T;P."JE HEBT DE HIGHSCORE VERBETERD !";T=U
 950 P."SCORE      : "U''"highscore : "T''"< DRUK TOETS OM TE H"
 960 ?#E1=0;P."ERSTARTEN >"
 970 LI.#FFE3;G.600
 980jF.Q=1TO2;PAUSE10;GOS.p;N.
 990 F.A=1TO6;UNSETA;N.;UNSET:HITC;CLEAR0;BLOCK1,10,5,44,20
1000 PRINT$30''''''''''''$9$9$9$9$9$9$9$9$9" H I G H W A Y "'
1010 MOVE11,25;DRAW11,36;L=0
1020 DRAW52,36;DRAW52,25;CIRCLE1,16,31,3;CIRCLE1,31,31,3
1030 CIRCLE1,46,31,3;MOVE31,36;DRAW31,40;CUBE1,0,21,40,20,
6;INV
1040 P.$30';?#E0=11;@=8;P.U;F.A=1 TO 3;DO IFA=1;D=#80A8
1050 IFA=2;D=#80AF
1060 IFA=3;D=#80B7
1070 IFA.R.%4=0;?D=49
1080 IFA.R.%4=1;?D=51
1090 IFA.R.%4=2;?D=55
1100 IFA.R.%4=3;?D=61
1110 U.?#B001=254;SOUND 255,255;N.
1120 IF?#80A8=?#80AF;IF?#80AF=?#80B7;G.1170
1130 F.Q=#FFTOOS.-10;SOUNDQ,10;N.;MODE4
1140 COPY#9000 #9200 #8000;O=0;M=M-2;IFM<0;M=0
1150 ?#13=0;?#14=0;IFP;F.X=1TOP;IMAGE:CAR3,(X*10-6),96;N.
1160 G.620
1170 F.Q=-100TO100S.2;SOUNDQ,5;INV;N.;INV;Q=0;IF?#80A8=49;U=
U+50
1180 IF?#80A8=51;U=U+100
1190 IF?#80A8=55;U=U+300
1200 IF?#80A8=61;U=U+500
1210 G.1040
1220pF.X=0 TO 9;SOUND25,200;PAUSE5;N.;R.
1230sF.X=0 TO 9;SOUND75,100;PAUSE5;N.;R.
```

N.B. Atomisten die (nog) geen joystick hebben kunnen dit spel toch spelen. Gebruik voor links de toets '3' en voor rechts de toets 'G'. Verander in regel 1110 '254' in '127' en gebruik de shifttoets i.p.v. de vuurknop. Veel Speelplezier ermee ! ! !

Voor de programmatuur: disk- en bandjesarchief.

Een MDCR operating systeem: DOS 2.

Het zal in de club wel algemeen bekend zijn wat een MDCR is: een mass-storage divice dat werkt met de kleine cassettes en waarmede naar gelang de grootte van de datablokken, per spoor zowat 64Kb kan opgeslagen worden.

Er circuleren binnen onze club, bij mijn weten, twee programma's die de MDCR (van Philips) aan het werk zetten; in hoofdzaak verschillen ze hierin van elkaar dat de ene werkt vanaf de B poort van een VIA, de andere gebruikt de A poort en een bit uit de B poort.

Dit houdt in dat wie aan de ingebouwde VIA een printer heeft hangen, best het eerste gebruikt; wie een tweede VIA installeert kan zowel het ene als het andere gebruiken.

Ongeacht de verschillen van beide programma's, hebben ze in feite het bestaande cassette-systeem ietwat "opgetild" door een directory van de tape bij te houden en daarmede een aantal bewerkingen te doen die het gewone cassette systeem natuurlijk niet kan. Zo wordt een programma "gezocht" op de tape, vooruit of achteruit, wat met de gewone COS niet gaat.

De grote stap vooruit van een hand bediende cassette naar een automatisch bestuurde MDCR is echter nog steeds een muizepasje in vergelijking tot de afstand die naar een DISK-systeem moet afgelegd worden.

Een MDCR-operating systeem.

De idee om een nieuw MDCR operating-system te gaan maken is gegroeid uit een daaraan voorafgaande idee: namelijk om met onze DOS een RAM-DISK system te maken.
Voor wie dit een nieuwe kreet is; een RAM-DISK behandelt een deel van het RAM geheugen als een DISK, m.a.w. men houdt daar een directory van bij en kan daar dan alles doen wat men met een DISK doet, alleen nog sneller omdat de mechanisch bepaalde access-tijd van de gewone disk nu wegvalt.
In het licht van dit opzet werd de hele DOS uiteengerafeld en werd er een source van heropgesteld.
Dit eerste reuzewerk liet toe daarna die delen eruit te gaan gebruiken die nuttig zijn voor de RAM-DISK en andere te verwaarlozen. Bij terug-assembleren moest er echter voor gezorgd worden dat alle oude routines op hun plaats bleven zitten omdat andere DOS utility's direkte sprongen naar deze adressen maken.

Het bleek (en dat is nogal logisch ook) dat de DOS in twee grote stukken kan opgedeeld worden: een eerste deel dat het werk met de file's organiseert en een twede deel dat de disk drive aanspreekt langs de controller om.

Dit twede deel werd veranderd om nu niet een Diskcontroller aan te spreken maar om in RAM te gaan werken.
Dat de stap naar het "aanspreken" van een MDCR nu voor de hand lag is klaar. Alleen moest deze stap nu nog gezet worden.

Even samengevat:

Aan de hand van de bestaande DOS en met behoud van al zijn mogelijkheden werd een systeem opgezet dat nu naar keuze ofwel met een RAM-gebied of met de MDCR omgaat net alsof het een diskdrive is.
Wie dus nog niet met een Disk werkte kan niet weten wat dit inhoudt; maar zij zullen al wel hebben vastgesteld tot hun ergernis, dat sommige programma's "voor disk" geschreven zijn, dat is nu verleden tijd, alles wat voor disk geschreven was kan nu op de MDCR.

INSTALLATIE

Om deze gemodificeerde DOS te installeren zijn er verschillende mogelijheden die bvb afhangen van wat U reeds heeft in uw ATOM. Vertrekkend van de "standaard ? ATOM".

HARDWARE:

U moet geheugen in RAM hebben vanaf #2000, op tot #9FFF.
U moet een EPROM in #Exxx kunnen plaatsen (of een prot. RAM); dat kan bvb op de schakelkaart.
U moet de VIA hebben en daarvan gebruikt U de B-poort: die zit op PL7 of de bus in uw rack als U PL7 naar buiten hebt gevoerd.
U kan natuurlijk ook een andere VIA (met andere adressering) met behulp van de duovia kaart op het rack zetten, en daarvan weer de B poort gebruiken.

U moet vanaf de B-poort interfacen naar de MDCR volgens het verbeterde Marchal systeem, ofwel volgens het hierbijgevoegde schema. Daarop vindt U ook de goede aansluitvolgorde.

SOFTWARE:

Is aan al het voorgaande voldaan, dan volstaat het om de code van de DOS.2 in de EPROM of de RAM op #Exxx te hebben om te kunnen werken, tikt u *DOS dan gaat uw ATOM over op dit operating systeem dat nu in RAM of op de MDCR gaat werken, net alsof het een diskette is.
Hoe krijgen we die code? Door het programma *MDCR te runnen!!
Maar hiervoor moet U al over een disk-drive beschikken en deze moet ook de vier sourcen MDCR op disk hebben staan; dat het gehele ding in SALFAA 2.0 is geschreven is ook niet verwonderlijk.
Wat wel duidelijk is: U komt er niet met deze source tenzij U beroep kunt doen op iemand die al over disk beschikt. Om deze

:
moeilikheid uit de weg te gaan is er reeds een file "AMDCR" beschikbaar om van uit #Exxx DOS.2 te draaien voor MDCR. Daarbij werd uitgegaan van de idee dat de bestaande via op zijn fabrieksadres nml #B800 gebruikt wordt en dat U als RAMdisk het gebied vanaf #7FFF naar beneden gebruikt.

OVER DE SOURCE EN HET HOOFDPROGRAMMA.

Het hoofdprogramma #MDCR vereist om de code voor de DOS." te leveren, dat op de disk vanwaaruit het gerund wordt, nog aanwezig zijn: de files ADISK, TDISK, MDCR1, MDCR2, MDCR3, MDCR4.
ADISK is de absolute-file van de gemodificeerde DOS.
TDISK is de symboltable benodigd door SALFAA.
MDCR1 is het begin van de source en bevat de gebruikte adressen.
MDCR2 is de source voor het MDCR besturingssysteem.
MDCR3 is het vervolg.
MDCR4 is het deel dat RAMdisk bevat.

AANPASSEN AAN INDIVIDUELE SITUATIES:

1: VIA adres: regel 105 van MDCR1 aanpassen aan het VIA-basis-adres

2: RAMdisk:wilt U hiervan afzien dan moet U in de file MDCR2 de regels 1006,1008,1010 weglaten; en in de file #MDCR moet U verder verwijderen de regels 1036 tot em met 1042.

3:Aantal MDCR's. Indien slechts een MDCR gebruikt wordt dan verandert men in file MDCR3 op regel 1180 LDX@#10 in LDX@#0 en laat verder regel 1178 weg.

Opmerking:
De basis-file "ADISK", die de gemodificeerde DOS bevat, herkent 16 mogelijke drivesl! Daarvan werden toegewezen nrs 0 en 1 aan de MDCR, nrs 2 en 3 aan RAMdisk. Wordt met opzet of per vergissing geprobeerd een "drive" te accessen die niet is aangesloten dan gaat het systeem "hangen".

4:Geheugengebied.
De code wordt door Salfaa in RAM op gebied #7000 neergezet en is geschreven voor #Exxx. Vanaf #7xxx kan het dan gecopieeerd worden bvb naar een EPROM.
Alhoewel het steeds mogelijk is om de code elders (dan in #Exxx) neer te zetten, bvb op de schakelkaart in #Axxx, en daar zal snel de bekoring toe komen, willen de auteurs erop wijzen dat dan aan het originele ATOM opzet afbreuk wordt gedaan. Wie echter zowel de originele DOS samen met dit systeem wil installeren, zal dan een soort schakelsysteem, dat beurtelings het ene en dan weer het andere voorzet, moeten ontwikkeen.

GEBRUIK.

:
Dit wordt echt kort gehouden: er bestaat een originele DOS handleiding, en een nederladse versie van TELEC, beiden zijn onveranderd bruikbaar voor dit systeem. Wanneer U geen eigen opstartprogramma (BOOT) heeft dan moet U *DOS intikken : alle vectoren worden nu verzet om te werken met DOS.2; na elke BREAK moet U terug *DOS tikken.

Zoals bij een gewone diskette moet U formateren. Dit gebeurt zowel voor de MDCR als voor de RAMdisk. Het format programma is, als bij de gewone dos als utility gedacht en staat als source code op de schijf: om het te runnen moet dit weer met Salfaa 2.0. De code kan komen waar U wil en dan roept U deze aan met LINK Xxxx om te formatteren.
Het formatteren van de RAMdisk is nog eenvoudiger: dit is een Basic programma dat U zet waar U wil (wel buiten de RAMdisk-space) en dat U dan runt. RAMdisk start op #7FFF en gaat vandaar naar beneden, U moet dus opgeven hoever; let er op niet in de buurt te komen van Top van een in geheugen staand programma want tussen beide is er geen beschermende barriere, zodat er ongelukken kunnen gebeuren.

TOT SLOT.

Uit wat voorafgaat kan afgeleid worden dat in dit systeem nog meer mogelijkheden zitten die we hier reeds willen aanstippen.
In deze DOS.2 zit een file en directorybehandelingssysteem dat, in princiepe, allerhande apparatuur kan aanspreken, daarom werd het geschreven om 16 "drives" te kunnen accessen.
Zoals we nu door het opgeven van een drivenummer, de keus hebben om twee mdcr en twee RAMgebieden t accessen, zal het mogelijk zijn om verder nog bvb 4 diskdrives achter dit systeem te hangen. Daarvoor moet dan wel het specifieke programma geschreven worden dat deze drives via hun controller aanspreekt. Dat kan dan elke controller zijn dus is het niet uitgesloten dat er vroeg of laat een controller voor een harddisk achter komt. Voor een harddisk zijn er echter nog andere problemen te klaren. Dat wordt dan wel DOS.3.
Het hele programma-werk van deze DOS.2 is van de hand van Michel; mijn bijdrage bestaat uit het leveren van deze tekst en misschien hier en daar een suggestie.
dit projekt beschouwen wij niet als ten einde, dat ziet U wel in geloven wij. Dat we nu toch publiceren is omdat we overtuigd zijn dat in dit stadium velen reeds nut kunnen hebben van het systeem. En anderen kunnen misschien nu inspringen om bepaalde ontwikkelingen over te nemen; bij voorbeeld:
De RAMdisk gebruikt nu eigen RAM van de ATOM. Men kan dit met weinig veranderingen zo maken dat de RAMdisk een eigen geheugenkaart bewerkt die dan een max. totaal aan geheugen van (16*256*256) 1.048.576 bytes kan hebben. Daarvoor moet echter een RAMdisk geheugenkaart met eigen decodering ontwikkeld worden, een hardprojekt dus.
Het schrijven van het programma om de controllers 8271 en 1793 te

besturen.
Het uitzoeken van een systeemuitbreiding nodig om een winchester controller in gang te krijgen.
Žoals men kan zien: onze ATOM is nog niet dood!

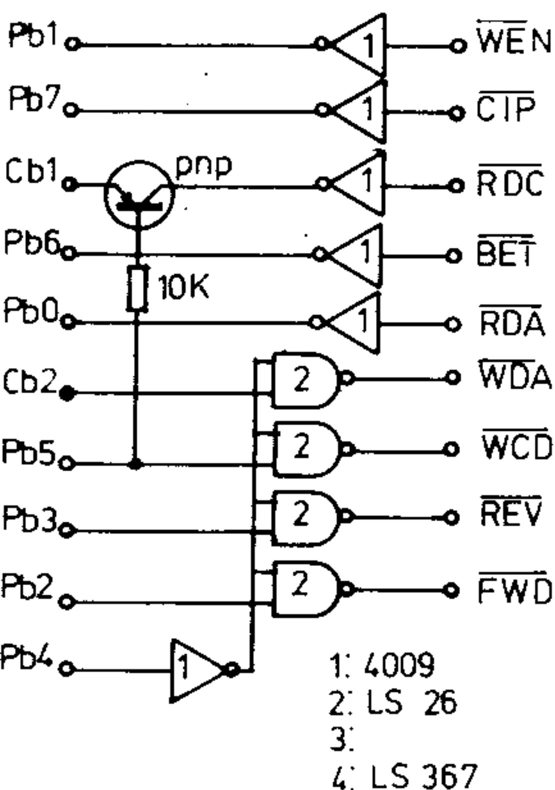
Michel Van Leuven en Jan Lernout A.C.B. 1 Maart 1987

INHOUD SCHIJF:

mdcr (tekstfile voor EDIT)
#MDCR
ADISK
TDISK
MDCR1
MDCR2
MDCR3
MDCR4
AMDCR
TMDCR
FORMAT
RFORMAT

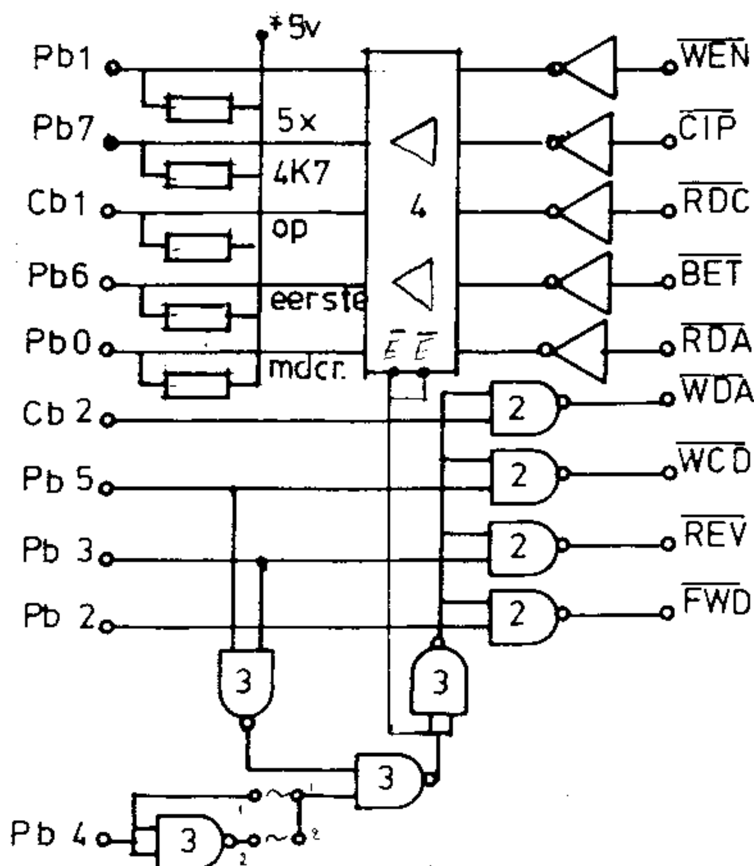
LITERATUUR:

ATOM DOS
Handleiding bij diskdrivepakket van TELEC
ATOM-WARE nr.2 van Henk Reinders
De DOS uitgedost door Frank Cuypers.



Eén MDCR

fig.1



Twéé MDCR's.

INLEIDING.

Toen enige tijd geleden voor de Acorn Atom de 80-koloms kaart ter beschikking kwam, is hij vrij snel in mijn computer gekomen.
Een toepassing, die ik er voor geschreven heb, is een memory-editor.
Het programma is geheel geschreven in SALFAA en kan als statement worden ingebouwd in een box of een P-CHARME-tabel.

BESCHRIJVING.

De routine wordt via de interpretortabel van de box aangeroepen, waarna gesprongen wordt naar het punt waar ook het adres wordt ingelezen dat achter het statement stond (zie regel 1940) en in de service-line worden de toetsen getoond waarmee de verschillende functies worden aangeroepen.
Vervolgens zet subroutine "SCHERM" voor de eerste keer 256 bytes op het scherm. Dit gebeurt in 16 regels van 16 bytes en de regels zijn als volgt opgebouwd:
Hexadecimaal adres van het begin van de regel, de hexadecimale inhoud van de volgende 16 bytes en tenslotte de ASCII-weergave van die 16 bytes.
Per keer wordt dus 1 pagina op het scherm gezet.
De cursor wordt gezet bij de hex-byte links boven.
De "READ'KEY"-loop (r.2000) scant het toetsenbord en springt, afhankelijk van de toets, naar de verschillende subroutines.
Omdat ik een numeriek toetsenbord aan de Atom heb, stuur ik de cursor met de getallen 8, 6, 2 en 4 in de verschillende richtingen over het blok van 16 bij 16 bytes. Komt de cursor bij de rechter zijkant, dan gaat de cursor naar het begin van de volgende regel, tenzij de cursor al rechtsonder staat. Een teruglopende cursor springt van de linker zijkant naar het einde van de vorige regel tot de cursor linksboven staat.
Naar de CHANGE-routine (r.2480) wordt gesprongen als de spatiebalk wordt ingedrukt. Staat de cursor in het hex-gebied, dan worden de eerstvolgende 2 toetsen ingelezen en getest op geldigheid (0 t/m 9 en A t/m F). Staat de cursor in het ASCII-deel, wordt 1 toets ingelezen. De zo verkregen nieuwe waarde wordt op de juiste plaats in het geheugen gezet en de hele regel wordt opnieuw geschreven, waarna de cursor weer op de zelfde plats wordt teruggezet.
De JUMP-routine (r.2620) verplaatst de cursor van het HEX- naar het ASCII-gebied en omgekeerd.
De NEXT- (r.2680) en PREV-routine (r.2720) zetten resp. de volgende en de vorige 256 bytes op het scherm.
Met "Q" tenslotte wordt de loop verlaten en het scherm in zijn oorspronkelijke vorm geinitialiseerd.

Berry Lam
05120-21258

****
* I N F O M A S T E R voor de 80-kolomskaart *
****

Sinds het verschijnen van de 80-kolomskaart zal er op diverse fronten druk gesleuteld worden aan programma's waarvoor die 80-kolomskaart in feite al jaren had moeten bestaan . Infomaster van A.Marchall is daarvan een goed voorbeeld . Op 80-koloms is het met deze "database" veel mooier werken.
Indien wij de 80-koloms clubkaart gebruiken , is het beschikbaar geheugen met 0.5 K uitgebreid van #8000-#8200 , omdat de 80-kolomskaart over haar eigen videogeheugen beschikt.
Welnu , hiervan heb ik gebruik gemaakt om het ZEER compact geschreven Infomaster enigszins te verfraaien en heb er een DOS-versie van gemaakt voor de 80-kolomskaart , INFOM80 genaamd.
Aan de werking van het programma is niet gesleuteld , ik vond dat er op de kwaliteit niets aan te merken valt en bovendien moeten reeds bestaande bestanden feilloos kunnen draaien . Alleen de lay-out is , naar mijn smaak , verbeterd en heeft een professioneler uiterlijk gekregen. Het programma is daardoor 1K langer geworden , de sorteer -en zoekroutines zijn dan ook verplaatst naar #2600 en deze worden bij het Runnen van INFOM80 vanzelf ingeladen.

Voor de volledige handleiding van INFOM80 verwijs ik U naar:
Atom Nieuws 3-2 mei 1984 (blauw) pagina 10 t/m 24
Atom Nieuws 3-3 (blauw) voor knutselarijen . Wees voorzichtig met dit laatstgenoemde artikel , mij is niet bekend aan welke versie van Infomaster ik heb gesleuteld en bovendien is er geen byte meer vrij!

Op de regioschijf vindt u:

```
INFOM80   #8000-#9FFF (basic)
IM.CODE   #2600-#27FF (ass.code)
```

We gaan alsvolgt te werk:

```
zet de Atom in de 80-kolomsmode
*LOAD INFOM80
?18=#80
END
RUN
```

Ik dacht hiermee een waardevol programma te hebben gedecoreerd met hetgeen het verdient en...... alweer een uitbreiding voor onze 80-kolomskaart !!

Nogmaals DIAL

Over automatische nummerkiezers is al vaker geschreven in A.N., en de schema's zijn er in vele soorten en maten.
Bij veel van deze schema's ontbreekt het printontwerp, wat voor nabouwen een belemmering kan zijn.
Ook dacht ik dat het eenvoudiger en goedkoper moest kunnen dan tot nu toe is gepubliceerd.
Het resultaat is een schema met 1 IC (7404) en vijf transistoren.
Om een telefoontoestel te simuleren zijn verder nog drie relais nodig en een trafo van ongeveer 1:1, maar dit is beslist niet kritisch.
In combinatie met de software biedt de schakeling de volgende mogelijkheden:

- Kiestoondetektie, maar ook bezettoon en verbindingstoon (het overgaan van de telefoon) worden gesignaleerd
- Aanroepen via een uitbreidingsstatement in P-Charme of rechtstreeks in machinetaal
- Het kiezen van een telefoonnummer volgens de methode van het impulskiezen (IDK)
- Printontwerp bijgeleverd op enkelzijdige eurokaart

De hardware

Voor het basisidee ben ik uitgeggan van het schema van RJ Bronsveld uit A.N. 86-3.
De hierin aangebrachte wijzigingen bestaan uit het aanpassen van de kiestoondetektieschakeling en een wijziging in de interfacing met de ATOM.
De schakeling kan in drie aparte delen gesplitst worden:
1: Het kiesgedeelte
2: De kiestoondetektie
3: De interface met de ATOM

Het kiesgedeelte heb ik ongewijzigd overgenomen uit het schema van de heer Bronsveld, zodat ik voor de uitleg van de werking hiervan naar zijn artikel verwijs, waarin dit uitstekend beschreven staat.

De kiestoondetektieschakeling bestaat uit twee transistoren, waarvan de eerste als versterker werkt, en de tweede als schakelaar voor de erachter geschakelde 7404.
De ingangsgevoeligheid van de versterker bedraagt ca 20 mV
Bij mij heeft de kiestoon die op de netlijn vd telefoon staat een amplitude van 300 mV, gemeten over het telefoontoestel.
Dit kan al naar gelang van oa de afstand tot de telefooncentrale wat meer of minder zijn, maar er blijft ten allen tijde meer dan voldoende over om de versterker aan te sturen, zelfs bij een minder geschikte trafo.
Om de versterker niet te oversturen is achter de trafo een trimmer van 1k geschakeld.
Om ook bezettoon en verbindingstoon te kunnen detekteren moet deze trimmer wel juist ingesteld zijn.
Het afstellen gaat als volgt:

Draai je eigen nummer (dat is altijd bezet)
Stel nu de trimmer zodanig in dat de led voor de kiestoondetektie meeknippert met de bezettoon, zonder flauw te blijven branden
Meestal zal de trimmer hiervoor vrij ver dichtgedraaid moeten worden.

De interface met de ATOM wordt gevormd door drie identieke schakelingen voor het aansturen van de relais, en een schakeling die in digitale vorm het uitgangssignaal van de kiestoondetktie naar de ATOM stuurt
Hierbij geldt: 0= kiestoon aanwezig
1= geen kiestoon

Het onderscheid tussen kiestoon, bezettoon en verbindingstoon wordt softwarematig gemaakt door de duur van het signaal te meten.

Bezettoon : 0.25 of 0.5 s aan/uit

Verbindingstoon : 1 s aan 4 of 5 s uit

Kiestoon : constante toon

De aansturing van een relais wordt ook gedaan met een nul, na een reset of bij het inschakelen van de computer zijn alle VIA-bitjes namelijk hoog, zodat de schakeling dan niets doet en het stroomverbruik minimaal is.

De software
Als de print wordt aangesloten op de B-poort van de ATOM-VIA kan het programma zonder meer worden overgenomen, als dit niet het geval is zullen de regels 220 en 230 aangepast moeten worden, door hier de juiste adressen in te vullen voor het dataregister (r 220) en het data-directionregister (r 230).
Het programma is geschreven in SALFAA versie 2.0, maar met oudere versies moet het ook werken.
Regel 10 schakelt SALFAA in (bij mij in voet 5).
Regel 20-50 maakt tabel voor P-Charme op #600.
Regel 60-80 assembleer programma
Regel 130 schakel P-Charme in (bij mij in voet 3).
Regel 150-250 bij een andere versie van SALFAA of om een andere VIA op te geven zal hier iets gewijzigd moeten worden.
Relocaten van het programma is mogelijk door regel 210 te wijzigen.

Het gebruikte statement is DIAL<string>
Voorbeeld: DIAL"010-4142434"
DIAL$A
Er worden door het programma twee zero-page geheugenlokaties gebruikt nl #B0 en #B1.
Deze ruimte wordt ook gebruikt door de DOS-ROM, maar dit heeft nog geen errors tot gevolg gehad.
Het gebruik ervan is tijdelijk, namelijk alleen tijdens het uitvoeren van het statement, en omdat de DOS-ROM deze adressen ook maar tijdelijk gebruikt geeft dit geen problemen, althans bij mij niet.
Adres #B0 wordt gebruikt als teller voor het meten van de lengte van

de van het aangeboden signaal (zie tabel).
Adres #B1 geeft nadere informatie over de laatste uitgevoerde opdracht (=laatst gebelde nummer).
De inhoud van #B1 is als volgt gedefinieerd:

- 0 Bezettoon gedetekteerd
- 7 Error, fout cijfer in de string
- 255 Verbindingstoon gedetekteerd

Het toevoegen van het karakter - veroorzaakt een sprong naar de kiestoondetektieroutine, door dit karakter weg te laten tussen het netnr en het abonneenr worden alle cijfers achter elkaar gekozen, het toevoegen van streepjes achter het nummer veroorzaakt even zovele extra sprongen naar deze routine
Na het laatste karakter uit de string springt het programma automatisch naar de routine.
Bezettoon wordt gedetekteerd na drie inpulsen, verbindingstoon na een puls van een seconde.
Het aantal pulsen dat nodig is om bezettoon te detekteren kan worden gewijzigd door de waarde van #B1 in regel 410 aan te passen, meer kan, maar kost meer tijd, minder is ivm storing niet aan te raden.
Tijdens het wachten op de kiestoon kan het programma beeindigd worden door op CTRL te drukken.
De inhoud van #B1 is nu te berekenen met de volgende formule: beginwaarde - aantal getelde pulsen, waarbij geldt dat een puls pas geteld wordt als hij voorbij is.
Om het on line schakelen van een modem te kunnen verzorgen is een grapje uitgehaald bij het konstateren van bezettoon.
Een modem moet namelijk op de uitgang van de kiezer aangesloten worden, parallel aan de telefoon.
Voor automatisch aanloggen moet het modem ook vooraf on line geschakeld worden, de kiezer draait dan het nummer, na het kiezen valt het relais af en het modem staat aangekoppeld.
Dit werkt zolang de lijn vrij is, maar als het gebelde nummer in gesprek is, en de kiezer zou normaal de lijn terugschakelen naar de ruststand, dan zou het modem aan een lijn hangen met bezettoon erop, en bij de volgende poging om het nummer te bellen zou de kiezer nooit geen kiestoon meer krijgen, want daarvoor moet eerst de hoorn erop gelegd zijn, wat dus overeen zou komen met het off line schakelen van het modem, opnieuw intikken van DIAL"....", modem online, wachten, weer bezet, lelijk woord, enz...
Wanneer de kiezer echter bezettoon detekteert, wordt de verbinding verbroken door het impulskontakt te openen en 1 seconde te wachten ipv de lijn meteen terug te schakelen.
Dit geeft de apparatuur in de telefooncentrale de indruk dat de verbinding verbroken is, en wanneer de kiezer dan opnieuw de lijn overpakt voor een nieuwe poging staat er weer keurig kiestoon op.
In een programma ziet dit er dan zo uit:

```
990$A="020-167383";REM ATOMTEL
1000DO
1010DIAL$A
1020UNTIL?#B1=#FF
1030LINK <startadres viditelprogramma>
1040END
```

Het geheel is bij mij al enige maanden in bedrijf en funktioneert nog

steeds naar behoren.
De totale bouwprijs wordt hoofdzakelijk bepaald door de relais, maar meer dan 30,- zal het niet snel worden, daar komt nog bij dat van de gebruikt onderdelen het meeste zo gangbaar is dat je het waarschijnlijk zo op de plank hebt liggen.

```
>  190 9000 10             .OPTION#10
   200 9000 6000 6FFF  .TABLE#6000,#6FFF
   210 9000 7000           .CODE#7000
   220 7000 B800          :drB=#B800
   230 7000 B802          :ddrB=#B802
   240 7000 FB83          :DELAY=#FB83       \ DELAY X/60 S
   250 7000 FFF4          :OSWRCH=#FFF4
   260 7000 20 B1 CE        JSR#CEB1
   260 7003 20 E4 C4        JSR#C4E4
   270 7006 20 23 70        JSR START
   270 7009 A0 00           LDY@0
   270 700B 84 B0           STY#B0
   280 700D A2 1E           LDX@30
   280 700F 20 83 FB        JSR DELAY
   290 7012               :LOOP
   290 7012 A4 B0           LDY#B0
   290 7014 B1 52           LDA(#52),Y
   290 7016 C8              INY
   290 7017 84 B0           STY#B0
   300 7019 C9 0D           CMP@13
   300 701B F0 11           BEQ STOP
   310 701D 20 83 70        JSR CALL
   310 7020 4C 12 70        JMP LOOP
   320 7023               :START
   330 7023 A9 70           LDA@#70
   330 7025 8D 02 B8        STA ddrB
   340 7028 A9 BF           LDA@#BF
   340 702A 8D 00 B8        STA drB
   350 702D 60              RTS
   360 702E               :STOP
   370 702E 20 3C 70        JSR TONE
   370 7031 20 ED FF        JSR#FFED
   380 7034               :EINDE
   390 7034 A9 FF           LDA@#FF
   390 7036 8D 00 B8        STA drB
   400 7039 4C 5B C5        JMP#C55B
   410 703C               :TONE
   420 703C A9 03           LDA@3
   420 703E 85 B1           STA#B1
   430 7040               :REPT
   440 7040 A2 00           LDX@0
   450 7042               :TEST              \CTRL PRESSED?
   460 7042 AD 01 B0        LDA#B001
   460 7045 C9 BF           CMP@191
   460 7047 D0 05           BNE CONT
   470 7049 68              PLA
   470 704A 68              PLA
   470 704B 4C 34 70        JMP EINDE
   480 704E               :CONT              \TOONDETECTIE BIT 7
```

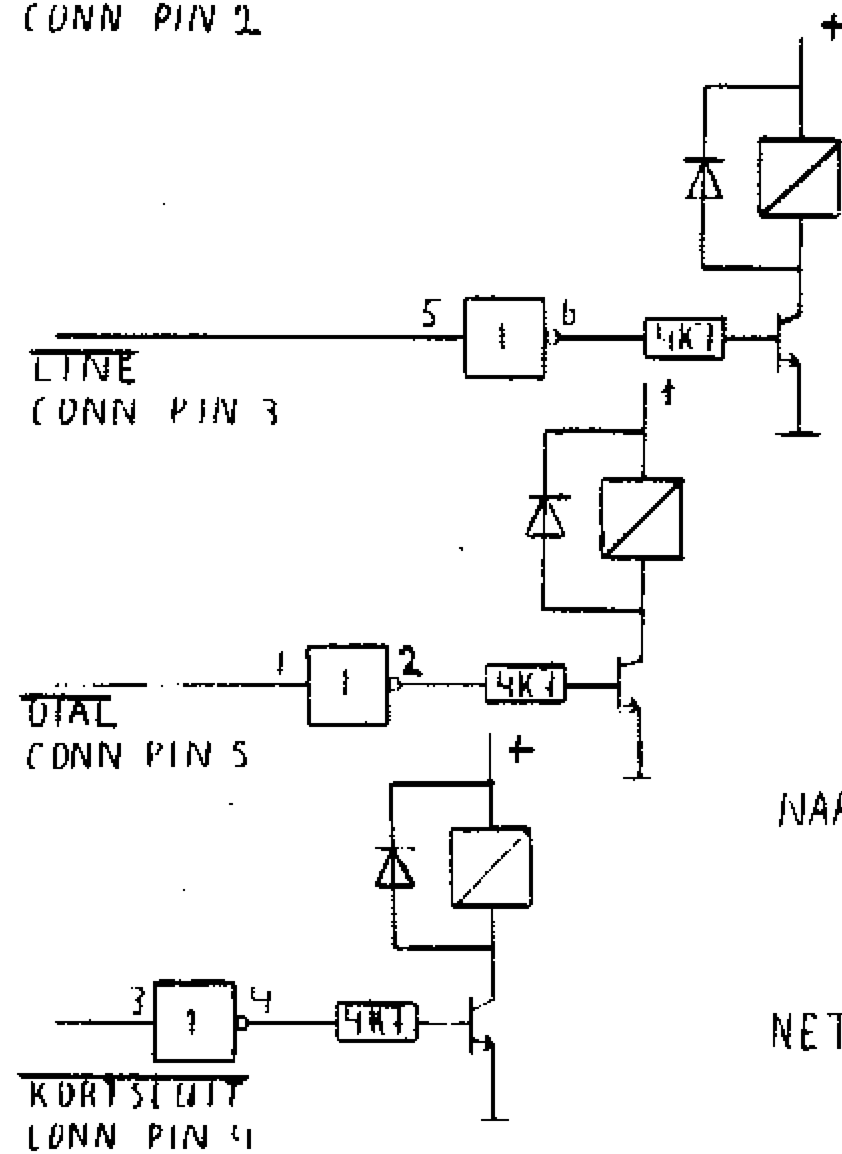
```
490 704E 20 66 FE    JSR#FE66
500 7051 2C 00 B8    BIT drB
500 7054 30 05       BMI NUL
510 7056 E8          INX
510 7057 E0 40       CPX@64
510 7059 D0 E7       BNE TEST
520 705B            :NUL
530 705B E0 0C       CPX@12
530 705D 30 E1       BMI REPT
540 705F E0 20       CPX@32
540 7061 30 09       BMI BEZET
550 7063 E0 32       CPX@50
550 7065 30 D5       BMI TONE
560 7067 E0 3F       CPX@63
560 7069 30 13       BMI CONNECT
570 706B 60          RTS
580 706C            :BEZET
590 706C C6 B1       DEC#B1
590 706E D0 D0       BNE REPT
595 7070 AD 00 B8    LDA drB
595 7073 29 EF       AND@#EF
595 7075 8D 00 B8    STA drB
596 7078 A2 3C       LDX@60
596 707A 20 83 FB    JSR DELAY
600 707D 60          RTS
610 707E            :CONNECT
620 707E A9 FF       LDA@#FF
620 7080 85 B1       STA#B1
630 7082 60          RTS
640 7083            :CALL
650 7083 20 F4 FF    JSR OSWRCH      \SCREEN ECHO
660 7086 C9 2D       CMP@#2D
660 7088 D0 04       BNE CIJFER
670 708A 20 3C 70    JSR TONE
670 708D 60          RTS
680 708E            :CIJFER
690 708E 29 0F       AND@#0F
690 7090 C9 0A       CMP@10
690 7092 10 40       BPL ERROR       \GETAL GROTER DAN 9
700 7094 C9 00       CMP@0
700 7096 D0 03       BNE KIEZEN
710 7098 18          CLC
710 7099 69 0A       ADC@10
720 709B            :KIEZEN
730 709B A8          TAY
730 709C AD 00 B8    LDA drB
730 709F 29 DF       AND@#DF
730 70A1 8D 00 B8    STA drB
740 70A4 A2 0F       LDX@15
740 70A6 20 83 FB    JSR DELAY
750 70A9            :reB
760 70A9 AD 00 B8    LDA drB
760 70AC 29 EF       AND@#EF
760 70AE 8D 00 B8    STA drB
770 70B1 A2 03       LDX@3
770 70B3 20 83 FB    JSR DELAY
780 70B6 AD 00 B8    LDA drB
780 70B9 09 10       ORA@#10
780 70BB 8D 00 B8    STA drB
```

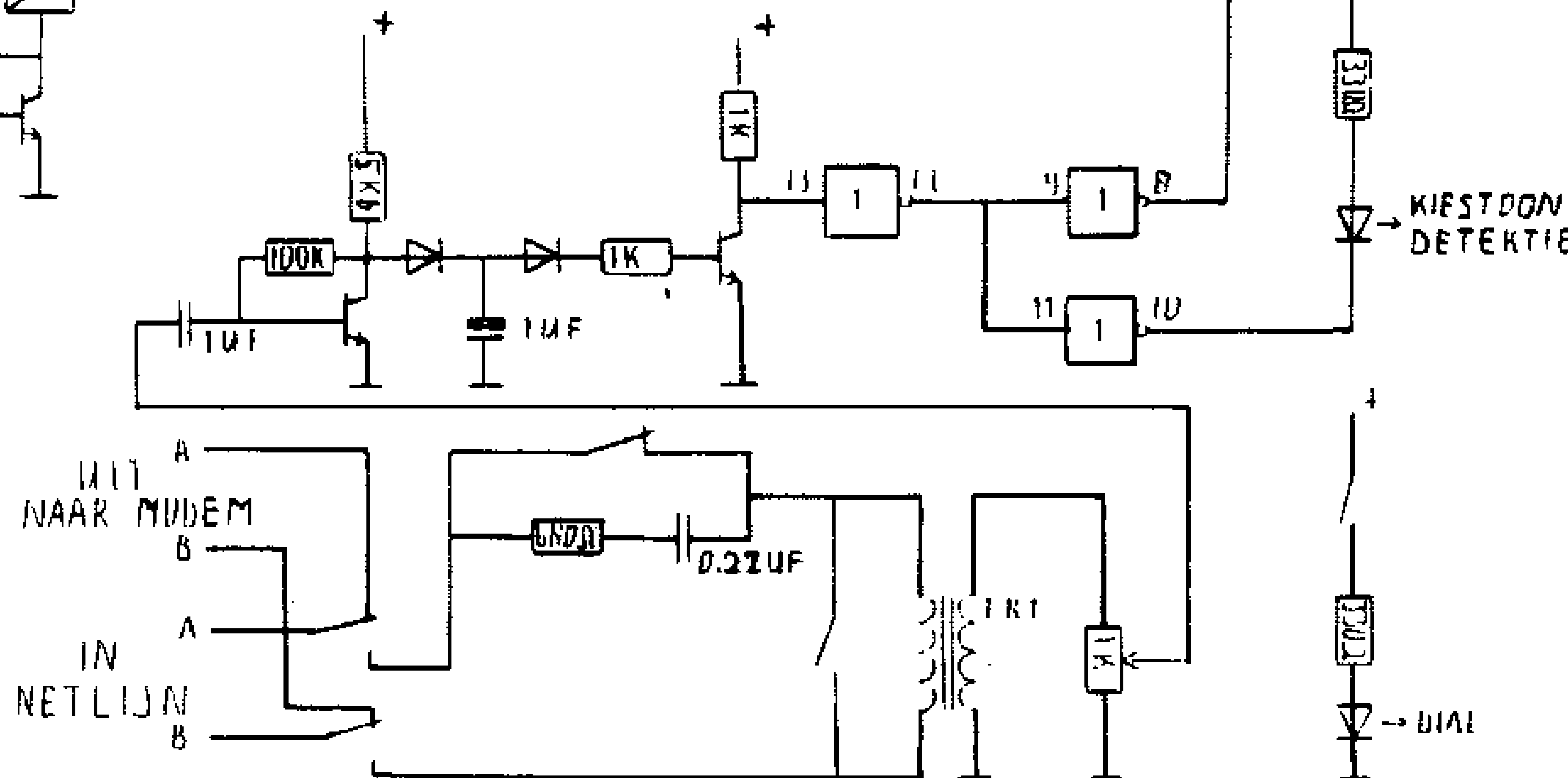
```
790 70BE A2 03        LDX@3
790 70C0 20 83 FB     JSR DELAY
800 70C3 88           DEY
800 70C4 D0 E3        BNE reB
810 70C6 A2 0F        LDX@15
810 70C8 20 83 FB     JSR DELAY
820 70CB AD 00 B8     LDA drB
820 70CE 09 20        ORA@#20
820 70D0 8D 00 B8     STA drB
830 70D3 60           RTS
840 70D4             :ERROR
850 70D4 A9 07        LDA@7
850 70D6 85 B1        STA#B1
850 70D8 20 F4 FF     JSR OSWRCH
850 70DB 4C 34 70     JMP EINDE
```

```
 10?#BFFF=5
 20?#3FC=6
 30!#600=#C6E3FF
 40$#603="DIAL"
 50!#607=#8000F0
 60ASM-V
 70PASS2;GOS.150
 80PASS1;GOS.150
130?#BFFF=3
140E.
150ASM-B
160.HARD
180.PRINT 27,78,10
190.OPTION#10
200.TABLE#6000,#6FFF
210.CODE#7000
220:drB=#B800
230:ddrB=#B802
240:DELAY=#FB83 \ DELAY X/60 S
250:OSWRCH=#FFF4
260JSR#CEB1;JSR#C4E4
270JSR START;LDY@0;STY#B0
280LDX@30;JSR DELAY
290:LOOP LDY#B0;LDA(#52),Y;INY;STY#B0
300CMP@13;BEQ STOP
310JSR CALL;JMP LOOP
320:START
330LDA@#70;STA ddrB
340LDA@#BF;STA drB
350RTS
360:STOP
370JSR TONE;JSR#FFED
380:EINDE
390LDA@#FF;STA drB
400JMP#C55B
410:TONE
```

```
420LDA@3;STA#B1
430:REPT
440LDX@0
450:TEST    \CTRL PRESSED?
460LDA#B001;CMP@191;BNE CONT
470PLA;PLA;JMP EINDE
480:CONT    \TOONDETECTIE BIT 7
490JSR#FE66
500BIT drB;BMI NUL
510INX;CPX@64;BNE TEST
520:NUL
530CPX@12;BMI REPT
540CPX@32;BMI BEZET
550CPX@50;BMI TONE
560CPX@63;BMI CONNECT
570RTS
580:BEZET
590DEC#B1;BNE REPT
595LDA drB;AND@#EF;STA drB
596LDX@60;JSR DELAY
600RTS
610:CONNECT
620LDA@#FF;STA#B1
630RTS
640:CALL
650JSR OSWRCH \SCREEN ECHO
660CMP@#2D;BNE CIJFER
670JSR TONE;RTS
680:CIJFER
690AND@#OF;CMP@10;BPL ERROR   \GETAL GROTER DAN 9
700CMP@0;BNE KIEZEN
710CLC;ADC@10
720:KIEZEN
730TAY;LDA drB;AND@#DF;STA drB
740LDX@15;JSR DELAY
750:reB
760LDA drB;AND@#EF;STA drB
770LDX@3;JSR DELAY
780LDA drB;ORA@#10;STA drB
790LDX@3;JSR DELAY
800DEY;BNE reB
810LDX@15;JSR DELAY
820LDA drB;ORA@#20;STA drB
830RTS
840:ERROR
850LDA@7;STA#B1;JSR OSWRCH;JMP EINDE
860.END
870R.
```

TRANSISTOREN	5 X BC547B
DIODEN	DUS
IC1	74LS04
TR1	TRAFO 1:1


KIESTOONDETEKTIE UIT
CONN PIN 2
LINE
CONN PIN 3
DIAL
CONN PIN 5
KORTSLUIT
CONN PIN 4
4K7
+


5K6
100K
1UF
1K
UIT
NAAR MODEM
IN
NETLIJN
0.22UF
KIESTOON
DETEKTIE
DIAL
+

Wilt u lid worden van de ATOM COMPUTER CLUB?
Neem dan contact op met de penningmeester van de regio waar u bij ingedeeld wilt worden.Deze kan u inlichten omtrent het lidmaatschap.

Regio NOORD;
D.Uuldriks Tjamme 18 9642 JV Veendam
05987-19611

Regio OVERIJSSEL/GELDERLAND;
H.de Ruiter Polarisstraat 25 8303 AC Emmeloord
05270-17824

Regio TWENTE;
G.J.Noorland Prinses Ireneweg 4 7433 DE Schalkhaar
05700-25294

Regio NOORD-HOLLAND;
P.van Kuik Zuideinde 54-a 1843 JP Groot-Schermer
02997-1902

Regio DEN HAAG;
Th.WaayerL.Couperusstraat 6 2274 XP Voorburg
070-862504

Regio DELFT;
F.von Morgen Postbus 145 2600 AC Delft

Regio ROTTERDAM;
R.de Haan Brasem 125 2986 HA Ridderkerk
01804-25160

Regio CENTRUM;
P.van Mourik Ruiterstede 60 3431 XN Nieuwegein
03402-48781

Regio ARNHEM;
J.Hartog Keyenbergseweg 60 6871 WK Renkum
08373-13757

Regio ZEELAND;
E.Gijssel Ruysdaalstraat 6 4462 AD Goes
01100-32557

Regio BRABANT-OOST;
P.Ehrlig Roostenlaan 266 5644 BS Eindhoven
040-114183

Regio LIMBURG;
A.van Zantvoort Op het Kuilken 16 6067 AK Linne
04746-5146

Regio BELGIE;
R.Leyssens Oude Baan 127 3550 Heusden Belgie

Bij het aangaan van het lidmaatschap kunt u de contributie overmaken op de rekening van de federatie.Vermeld hierbij uw volledige naam, adres en de regio waar u bij ingedeeld wilt worden.